2015

Developing and Testing a Novel Low Cost Force and Torque Sensor Suitable Robotic Applications

Daniel Sherman
University of Rhode Island, dan_sherman@my.uri.edu

DEVELOPING AND TESTING A NOVEL LOW COST FORCE AND TORQUE SENSOR SUITABLE FOR ROBOTIC APPLICATIONS

BY

DANIEL SHERMAN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

MECHANICAL ENGINEERING AND APPLIED MECHANICS

UNIVERSITY OF RHODE ISLAND

2015

MASTER OF SCIENCE

OF

DANIEL SHERMAN

APPROVED:

Thesis Committee:

Major Professor	Musa Jouaneh
	David Chelidze
	Richard Vaccaro
	Nasser H. Zawia DEAN OF THE GRADUATE SCHOOL

UNIVERSITY OF RHODE ISLAND
2015

ABSTRACT

A novel low cost force-torque transducer suitable for robotic applications, its development from design to fabrication, and its capabilities are described. The transducer is capable of measuring planar forces and torques. The sensors used in the device are force sensing resistors. A numerical model for this transducer is described to evaluate the equations for the output of the device given a force input. A general design model is created for future work and is capable of being extended to three dimensions.

Force and torque loadings at pre-determined magnitudes and angles are applied to evaluate the device's ability to accurately measure forces at any angle *theta* and magnitude *F* as well as any Torque *T*. Testing of the system reveals the performance as described by its accuracy and repeatability. The performance of the device successfully correlates the equations and the numerical model to the actual output of the device. Thus, a force-torque transducer was successfully designed, tested, and evaluated.

ACKNOWLEDGMENTS

I would like to thank my friends, family, and especially my girlfriend who have been supportive of my academic career at the University of Rhodes Island. I would like to thank my advisor, Professor Musa Jouaneh for all of his guidance throughout my Graduate Studies. I would like to thank David Ferreira, Joe Gomez, and Jim Byrnes for their help with anything hands on in the machine shop and in the mechatronics lab. I would also like to thank the entire Mechanical, Industrial and Systems Engineering faculty and staff at the University of Rhode Island.

TABLE OF CONTENTS

LIST OF TABLES

TABLE PAGE

LIST OF FIGURES

FIGURE PAGE

CHAPTER 1

INTRODUCTION

Robotics play an ever increasing and vital role in the manufacturing industry in the automated processes of today and tomorrow. From tooling operations such as deburring a part, to assembly and disassembly operations in electronic waste operations, robots are playing an ever increasing role in various automated operations [1]. Advancements in robotic technologies continue to lower the cost associated with robotics.

Through various robotic operations, including tooling, assembly, and disassembly to name a few, there is often a need to measure the forces and torques applied to the end effector of a robotic arm. The need for this stems from the need to protect a robotic arm and the tooling from damage caused by a lack of force feedback [2-3]. Another area of concern is in precision operations, where care must be taken so as not to damage the part with which the operation is taking place. Typically, a robotic arm will locate a part with a known position and orientation and use position as a means of clamping down on the part. If the part is not in the correct position, the risk of damaging the part is greatly increased. Developing a force sensor for the end effector will provide a means of analyzing the force output and ensuring there is no damage to the end effector. The importance of lowering costs associated with robotics is that it lowers the overall cost associated with manufacturing. Another application where position based robotics is an issue is the possibility of a robotic arm applying too much load caused by interferences in the environment and workspace. Using position feedback alone could result in the robotic arm to become damaged.

The force and torque sensor in this particular study is a device attached to the end of a robotic arm and the end effector. It is positioned between the two, whereas the robotic arm is attached to one side, and the end effector is attached to the opposite side. By positioning it here, one can isolate and determine the forces and torques applied to the end effector.

Currently, there exist force and torque sensors on the market that effectively and accurately measure forces and torques. Perhaps the most common technology associated with these sensors is the use of a foil strain gage. A foil strain gage is a device which produces a change in resistance under strain. Through the mechanical properties of the transducer and associated equations, the applied forces and torques can be determined.

Current force and torque transducers available on the market that utilize strain gage technology suffer from a number of drawbacks, including the high cost associated with manufacturing the transducer and the high sensitivity to electrical noise and temperature change [3]. The high cost of the sensor arises from the difficulty and accuracy required in manufacturing the transducer. Another disadvantage is the maximum allowable strain is very close to the strain gage breaking point. This requires mechanical stops in order to limit the deflection of the stage and flexures associated with the design. The strain gages also require a difficult and time consuming process in order to attach them to the surface of the stage, which is a flexible part of the sensor whereas deflection of the stage causes a change in resistance of the gage. The sensitivity to electrical noise is also a problem as it effects the ability to accurately measure the value.

Force Sensing Resistors have a few advantages over strain gages that greatly reduce the costs associated with manufacturing force and torque transducers. The preferred method of attaching force sensing resistors is to use a thin polymer strip of double sided tape [4]. When compared to strain gages, this is a significant improvement as it reduces a significant amount of time to manufacture the device. This is also important in that if a sensor is to become damaged, replacing the sensor is a very simple task. The FSRs also do not require a stage to be manufactured to high tolerances. This is due to the fact that the location of the force sensing resistors is not in the same configuration as the strain gages.

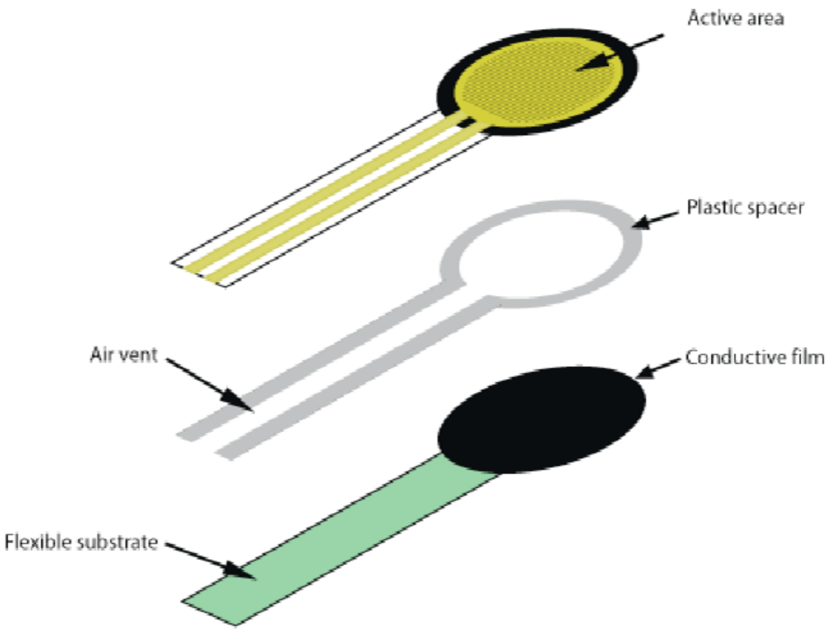


Figure 1: Force sensing resistor schematic [5]

The main sensor which will be researched and implemented is the force sensing resistor. A force sensing resistor is a thin film pressure transducer whose resistance changes as a function of pressure applied [4-6]. A schematic of a force sensing resistor is shown in Figure 1. The key components of the force sensing resistor are shown in the figure. These include an active area, a conductive film with a flexible substrate attached to the end, and a plastic spacer with an air vent separating the two layers of film [5]. When the sensor is pressed on, the resistance increases predictably. The sensor itself is generally on the order of two to three tenths of a millimeter thick. It consists of a conductive polymer separated from a sensing film by a nonconductive polymer ring. As pressure is applied, the two come into contact with each other and the resistance value decreases logarithmically.

The importance of developing a low cost transducer is that it will greatly reduce the complexity and costs associated with strain gage transducers. Reducing the cost of the transducer will reduce the cost of fabricating a robotic workcell. Little research has been conducted on the use of force sensing resistors to measure forces and torques, let alone specifically for a robotic application. A novel method of sensing forces and torques is being presented in this paper. The use of a force-sensing resistor to measure the forces applied is being studied.

In this study, a method for measuring planar forces and torques will be developed and tested. Force sensing resistors will be secured to an inner stage. An outer ring will be placed around the stage and will apply a preload to the sensors. Because the sensors can only measure forces in compression, a preload will be applied to each sensor to ensure that the sensor can measure forces in tension as well. Each sensor will initially have an equal preload which will be equal to half the force range of the sensors used. For example, if a sensor has a force range of 100 Newtons, a preload of 50 Newtons will be applied. This will allow the sensor to read forces on the range of -50 Newtons to 50 Newtons. If effective, the device will allow for a cost effective method of measuring planar, three degree of freedom forces and torque. The design model created will serve as the building block to design and analyze a three dimensional, six degree of freedom sensor capable of measuring forces and torques in all directions.

The research conducted in this study has three main aims. The first specific aim is to determine if force sensing resistors can in fact be used to accurately measure forces and torques. Force sensing resistors have been used in past for measuring forces applied. Force sensing resistors have been shown to be effective in determining the forces applied in both human and prosthetic applications[7-8]. They have also shown to be effective for measuring forces in a preloaded condition [9-10].

The second specific aim of this research is to create a design model which can be used to design and set up experiments for this device. The design model will include all of the necessary kinematic equations that describe the device and will aid in selecting a sensor configuration and orientation to meet a set of force and torque specifications. The design model will be used to design the prototype for testing, as well as future work in the area.

The third specific aim of this research is to demonstrate the capabilities of this device. These capabilities will be defined by three parameters, namely accuracy, repeatability, and cost. The accuracy of the device will be based on the equations associated with the device and how well the equations correspond to the actual results measured from testing. The repeatability will be used to validate the effectiveness of the sensor. The cost to manufacture the device will justify the competitiveness of the device.

The manuscript is organized as follows. First, a full in-depth review of the literature available including past research and relevant patents is presented in Chapter 2. This will ensure that the research is novel and will give insight into how others have researched this topic. The proposed design for a force-torque sensor is presented in Chapter 3. Both the design equations and numerical model are presented as well. The fabrication of the prototype is presented in Chapter 3 and its testing results are presented in Chapter 4. Chapter 5 describe the conclusions drawn as well as describes future research.

CHAPTER 2

REVIEW OF LITERATURE

A full literature review and patent search was conducted at the beginning of the study in an effort to gain insight into previous research in the area as well as relevant patents. Doing so aided in directing the study. Two key components of the design of the transducer became apparent through the study, both of which will be elaborated on in Chapter 3. The first is that three sensors are not enough to uniquely measure the forces and torques applied to the transducer. The second is that the FSRs need to be compressed before being loaded.

There are numerous force and torque sensors available on the market. Many patents were analyzed that measured both force and torque, and many were very similar in design. One in particular, a patent [11] held by the Sony Corporation, gave insight into the general design of force and torque sensors available on the market. A figure of it is taken from the US patents website and is shown in Figure 2. The construction consists of a center stage, connected to the outer ring by four beams. Each of these beams has strain gages attached on each side. When forces and torques are applied to the center stage, the deflection of the beam strains the strain gage. The resistance of the gage changes. A change in voltage across the gage is measured, and is converted to a force. From there, the forces and torques acting on the system can be determined.

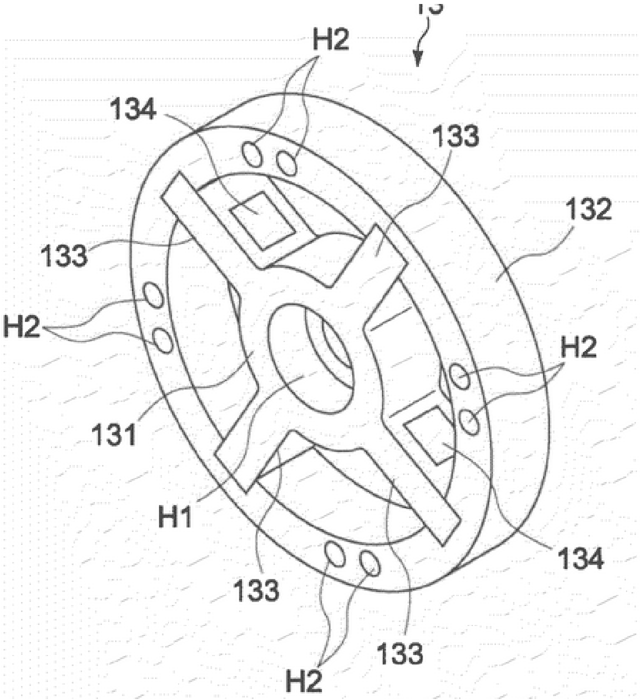


Figure 2: Force-Torque sensor using strain gages [11]

In sifting through the numerous patents available, one in particular became the inspiration for the method of sensing forces using a force sensing resistor. This patent provides a method of using a force sensing resistor in order to measure the power applied to the pedal of a bicycle through a show [9]. When using a pedal in which the user’s foot is secured and thus able to apply both a downward or positive force and an upward or negative force, there is a need to be able to measure both positive and negative forces. In order to measure both positive and negative forces, the sensor is pre-loaded to a force that is half of the measurement capability. By doing so, the shoe is able to measure in the positive region and in the negative region, while still applying pressure to the pedal. A figure of the pre-loaded show is shown in Figure 3.

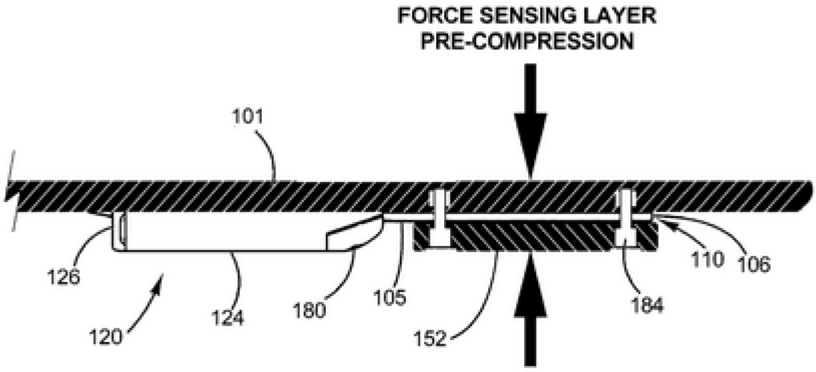


Figure 3: Pre-loaded FSR in a bicycle shoe [9]

This patent is particularly important for a few reasons. The first is that it presents a method for measuring both positive and negative forces using a single pre-loaded FSR. This is proof that a pre-loaded FSR can be accurately used to measure forces. This is also important as it only measures forces in one dimension. This research paper aims to extend the number of dimensions to two.

A work of literature which directly related to preloading sensors in a configurations suitable for measuring forces and torques is titled “Active Design Method for the static characteristics of a Piezoelectric Six-Axis Force/Torque Sensor” [12]. In this method, the sensors used were piezoelectric crystals. While these are not the sensors being used in this particular study, it gave insight into a force-torque sensor where pre-loaded sensing elements were used. Figure 4 shows the configuration of the pre-loaded elements.

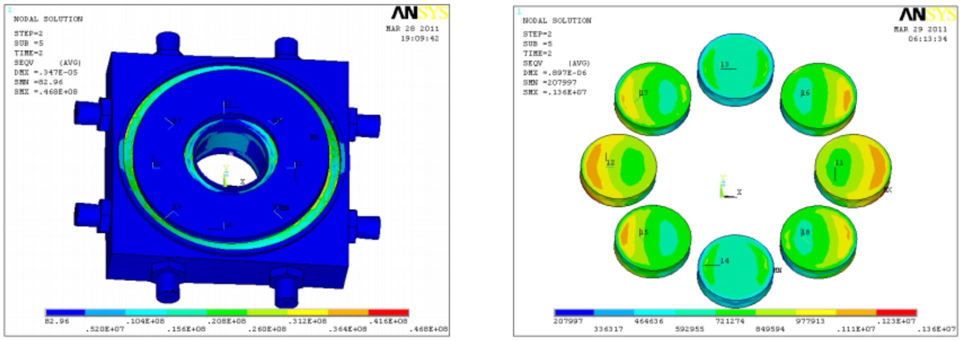


Figure 4: Piezo-Electric Force-Torque sensor [12]

A paper published by Schumacher and Jouaneh [1] demonstrates the development of a tool used for the disassembly of electronic components using FSRs. This device uses a tool on a pivot which is capable of measuring forces. The tool is shown in Figure 5.

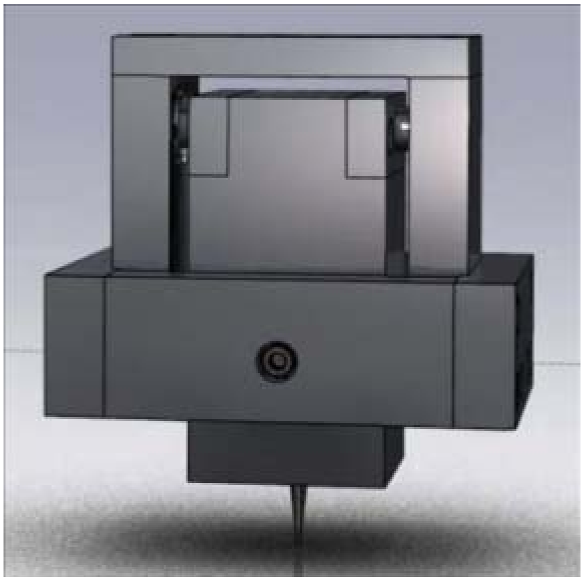


Figure 5: Tool for disassembly utilizing FSRs [1]

The tool has been designed to use three FSRs in order to measure forces in both the positive and negative x direction as well as normal to the tool tip. The tool works by pivoting on a bushing in order to apply forces to a FSR in the positive x direction and another FSR in the negative x direction as well as in the y direction in order to apply forces to the FSR normal to the tool. There are a few important things that can be taken from this. The first of these is that this paper demonstrates a method to accurately measures forces applied using FSRs. This is an excellent starting point in developing a force-torque sensor using FSRs, as this proves the ability of FSRs to be used to measure forces. However, the design of this tool not sufficient for designing a force-torque sensor. This paper provides a method using a tool that is not static. In order to create a usable force and torque sensor, the work reported in this thesis develops a sensor which is rigid in order to eliminate movement between the end effector and the robotic arm.

CHAPTER 3

METHODOLOGY

The theoretical groundwork for this study began by analyzing the method in which to transfer all forces applied to the individual sensors. In order for the transducer to effectively measure the forces and torques, the transducer needs to accurately, repeatability, and predictably measure all forces applied and in order for the transducer to do this, each individual sensor must also be able to. Initially, the main focus was on how to apply the forces to each of the sensors while using the least possible number of sensors necessary.

It was determined in the literature review that due to the nature of the force sensing resistor, in order to measure both positive and negative forces, the FSR would have to be under a preload. It was also determined that in order to measure torques applied at a given direction, each sensor would need to be positioned at an offset from the center. By doing so, each sensor would transmit a positive force in a certain direction of force, and a negative force in the opposite directions.

One of the main differences between a strain gage and a force sensing resistor, and what proved to be one of the biggest challenges to overcome, is that a force sensing resistor can only work in compression. As a result of this, the force sensing resistor must be preloaded. By preloading the FSR to a value of P preload, this allowed the forces to be applied in both directions, as applying a force in one direction would increase the value of the FSR reading a value higher than *P*, and applying the load in the opposite direction would decrease the load to a value less that *P*.

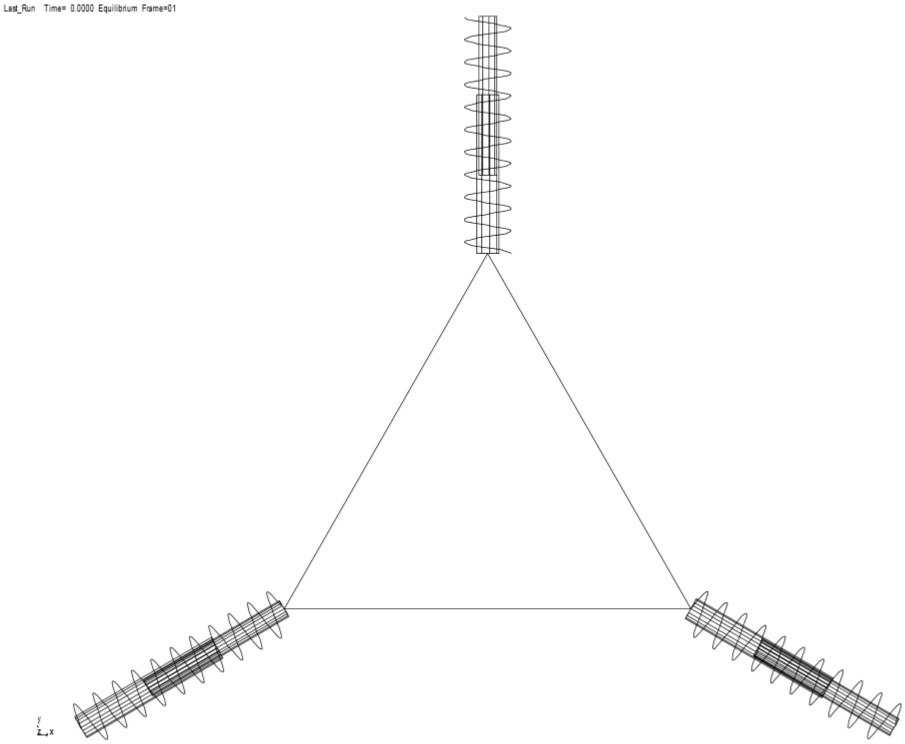
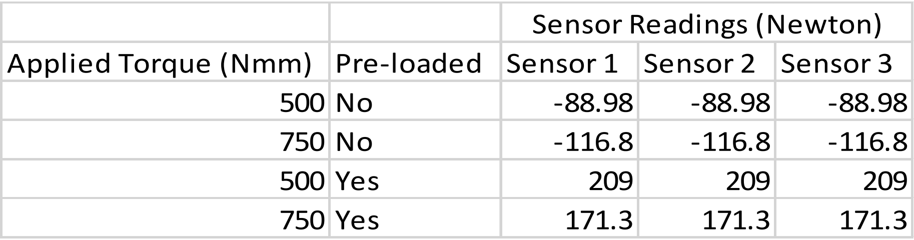


Figure 6: Three FSR transducer preliminary model

In order to gain greater insight into what would happen when each of the sensors were preloaded and how they would react, a model was created where a triangular center stage was constrained symmetrically by three springs connected to the ground, as shown in Figure 6. In the model, the triangular stage was constrained by planar motion, only being able to undergo translation in the x and y, and rotation around the z axis. Each of the springs were considered linear, having equal spring constant *k* and free length l_0. The springs represented the force being applied to an FSR, whereas the force in the spring would directly press on the sensor and transfer all force to the FSR. In this model, the actual shape of the center stage was arbitrary and only the location of the spring connections were important because the stage was considered massless in order to simplify the model.

The model was translated to Adams/View and a simulation was run in order to get a better understanding of why preloading the sensors was necessary as well as to understand the difference in how the stage reacted when not preloaded as opposed to being preloaded. For this simulation, each spring was given a spring constant *k* equal to 1000 N mm. The preloaded sensors were preloaded to 220 Newtons. The preload was chosen due to the fact that the sensors that were going to be used in the actual device were 445 Newton sensors. Table 1 shows the results of each of the two tests. It is important to note that anywhere there is a negative force readout, the actual force readout of an FSR would in fact be 0, as it is impossible to read a negative force with an FSR. A few important conclusions can be drawn from this. The first of these is that the change in force of each sensor is independent of whether the sensor is preloaded. This holds true as long as the force is much smaller than the apparent preload of the sensor.

Table 1: Simulation results for pre-loaded and non-pre-loaded transducer



		Sensor Readings (Newton)		
Applied Torque (Nmm)	Pre-loaded	Sensor 1	Sensor 2	Sensor 3
500	No	-88.98	-88.98	-88.98
750	No	-116.8	-116.8	-116.8
500	Yes	209	209	209
750	Yes	171.3	171.3	171.3

According to the paper by Su, June, and McCarty, [13] given known spring lengths, there can be multiple real solutions to the problem. A second preliminary simulation was run in order to prove this. In this simulation, the previous values of *k* and *P* were used. Figure 6 shows the initial, unloaded sensor as a reference for the initial configuration. Figure 7 and Figure 8 display what happens when a torque of magnitude *M* equal to 2000 N mm is applied to the stage in the positive z direction compared with what happens when it is applied in the negative z direction. The results, shown in

Table 2, show that the sensors would in fact output identical readings even though different directions of torque have been applied. This is due to the fact that the change in force applied to each sensor is equal for both the positive and the negative direction cases. This is true for not only this particular case, however it has been proven in the journal article and no further proof is necessary.

Table 2: Three Sensor Pre-Loaded Transducer with positive and negative applied torque

		Sensor Readings (Newton)		
Applied Torque (Nmm)	Direction	Sensor 1	Sensor 2	Sensor 3
500		209	209	209
750		171.2	171.2	171.2
-500		209	209	209
-750		171.2	171.2	171.2

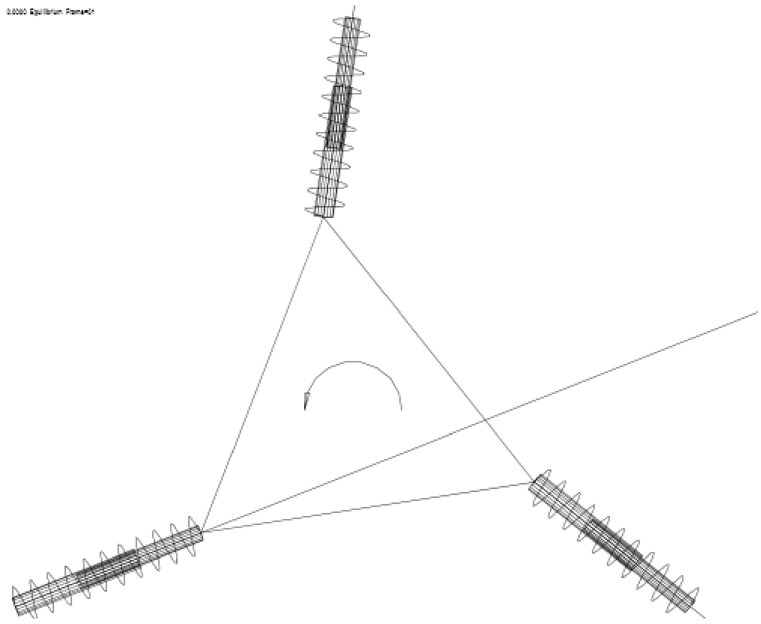


Figure 7: Three FSR transducer model with a positive torque applied

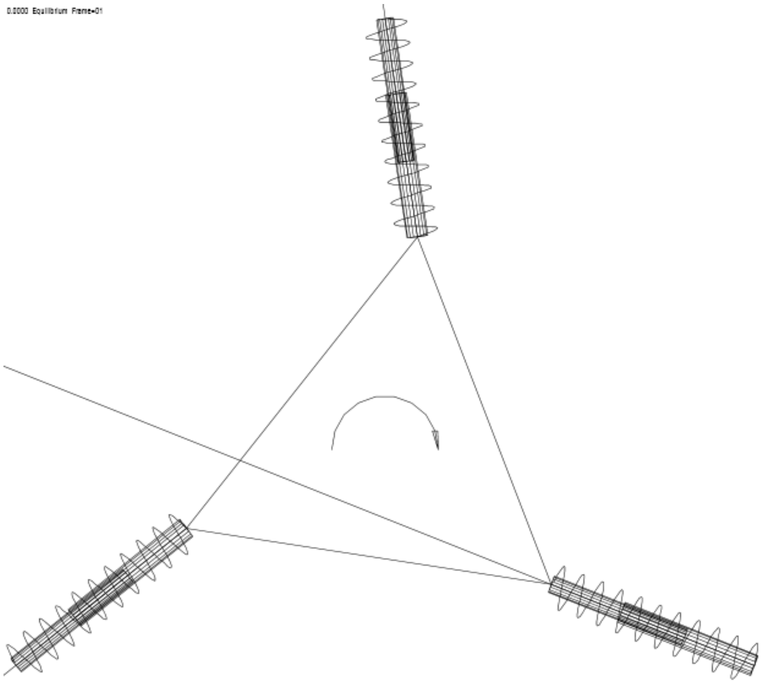


Figure 8: Three FSR transducer model with a negative torque applied

In order to solve this problem, two changes had to be made to this model. The first is that the sensors could not point directly at the center of the device. In this previous model, the sensors pointing at the center specifically created rotational symmetry. To solve this, the line of action of the sensors were all moved a distance *d* away from the center of the device. If each of the sensors were moved in the same direction, the transducer would not be able to maintain preload and the sensors would unload in order to reach equilibrium. To solve this, a fourth sensor was added in the configuration seen in Figure 9. Each of the sensors were moved a distance *d* away from the centerline. By doing this, symmetry was kept, and torque could now be measured. Preliminary results showing the torque directionality associated with this configuration are shown in Table 3.

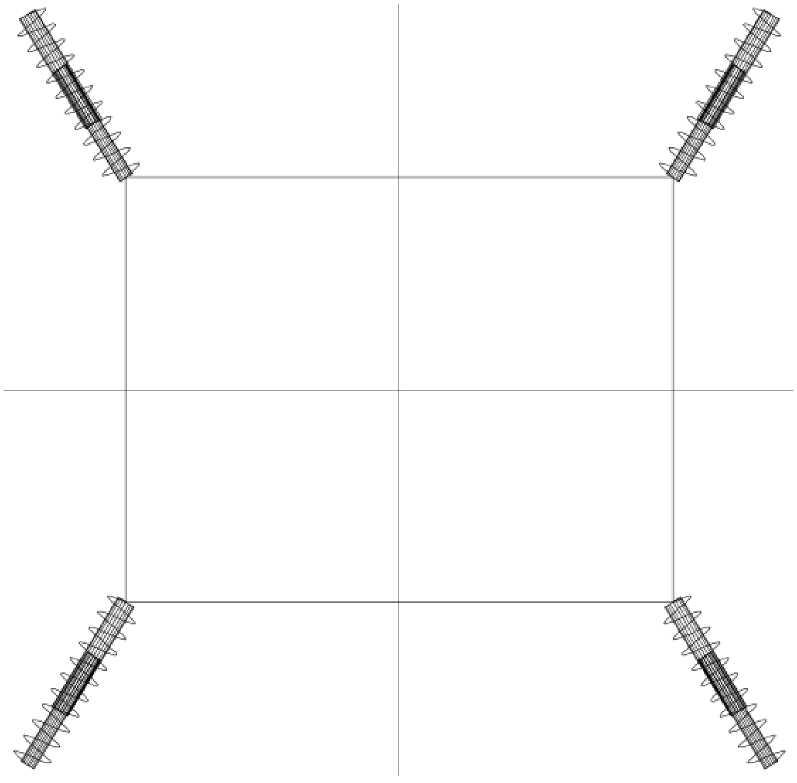
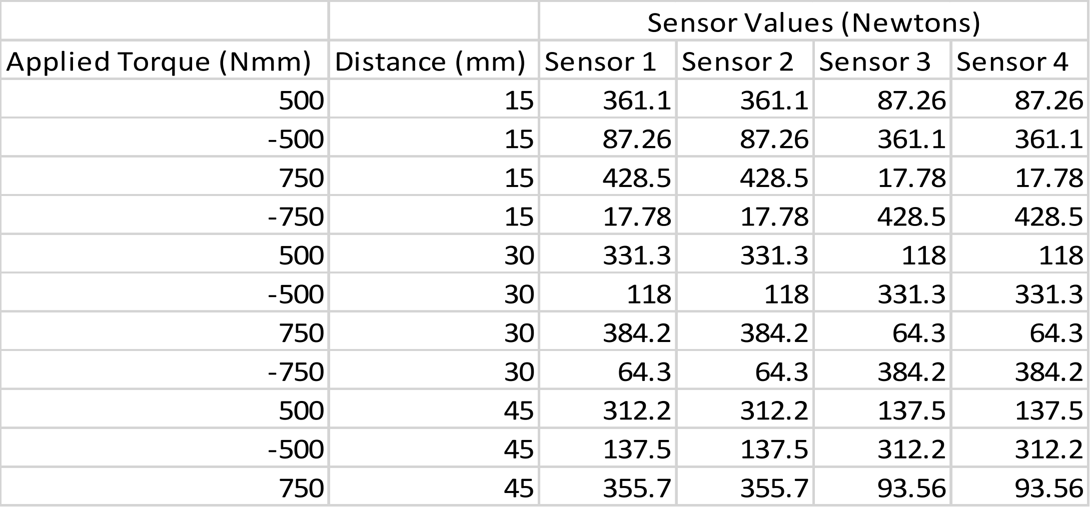


Figure 9: Four FSR transducer model

Table 3: Preliminary configuration results for four FSR transducer



		Sensor Values (Newtons)			
Applied Torque (Nmm)	Distance (mm)	Sensor 1	Sensor 2	Sensor 3	Sensor 4
500	15	361.1	361.1	87.26	87.26
-500	15	87.26	87.26	361.1	361.1
750	15	428.5	428.5	17.78	17.78
-750	15	17.78	17.78	428.5	428.5
500	30	331.3	331.3	118	118
-500	30	118	118	331.3	331.3
750	30	384.2	384.2	64.3	64.3
-750	30	64.3	64.3	384.2	384.2
500	45	312.2	312.2	137.5	137.5
-500	45	137.5	137.5	312.2	312.2
750	45	355.7	355.7	93.56	93.56

Significant research was spent on the kinematic equations associated with a stage supported by springs. In particular, the paper published in the ASME Journal of Mechanical Design titled “A Polynomial Homotopy Formulation of the Inverse Static Analysis of Planar Compliant Mechanisms” [13] proved to be the most informative work. The paper describes a method for determining the forces applied to a stage supported by *n* number of springs with known spring lengths and unloaded spring lengths using the principle of virtual work. In the case of the transducer, the pins used to transfer forces from the stage to the FSRs are modelled as springs of spring constant *k*, which is determined mathematically by the dimensions of the pin.

3.1 Principle of virtual work solution

To solve the system of equations for the model, the principle of virtual work was selected to translate the forces acting on each spring to a set of applied forces and torques. In this solution, each spring can be considered a revolute join connected to a prismatic, which is connected to another revolute joint. Each revolute joint allows rotation, and the spring can only extend or compress. The principle of virtual work is a well-documented solution to the planar mechanism and will be outlined in this section.

For convention, the position variables must all be determined. The location of the rigid pivots are specified at x and y coordinates A_i, and are taken to be grounded. The center stage is the moving body and is defined by equation (3.1), where B_i are the coordinates in terms of the ground and b_i are the coordinates in terms of the moving body. Because the moving body is rigid, b_i are all constants. C is the center of the stage given by an angle Θ rotated according to the right hand rule. All of these parameters are related by equation (3.1), which is simply a coordinate transformation to determine the B_i coordinates of the stage.

$$B_i = C + [R]b_i = \begin{bmatrix} C_x \ C_y \end{bmatrix} + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} b_{ix} \ b_{iy} \end{bmatrix}^T \qquad (3.1)$$

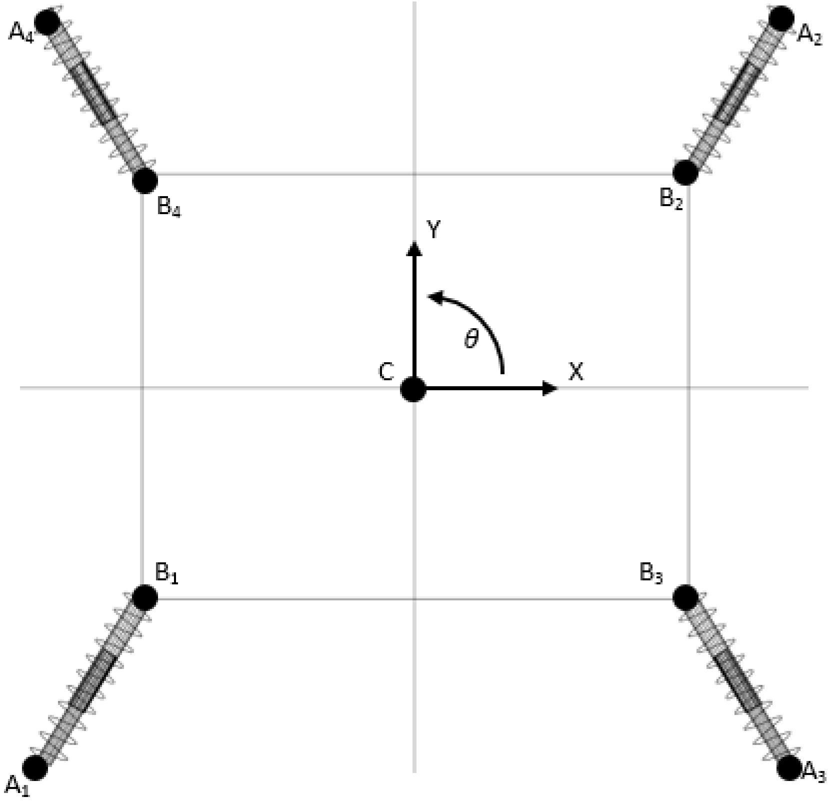


Figure 10: Principle of Virtual Work model of transducer

In the principle of virtual work equation (3.2), the virtual displacements of the stage, given as $\frac{\partial V}{\partial x}, \frac{\partial V}{\partial y},$ and $\frac{\partial V}{\partial \theta},$ are equal to the forces applied F$_x$, F$_{y,}$ T$_z$.

$$\frac{\partial V}{\partial x}\delta\mathrm{x} + \frac{\partial V}{\partial y}\delta\mathrm{y} + \frac{\partial V}{\partial \theta}\delta\theta = F_x\delta\mathrm{x} + F_y\delta\mathrm{y} + T_z\delta\theta \tag{3.2}$$

The change in potential energy of each spring is given as a function of the displacement of each spring by the following relationship, where k_i is the spring rate, l_i is the spring length, and l_i^0 is the free length of the spring.

$$\mathrm{V} = \sum_{i=1}^{3} \frac{1}{2} k_i \left(l_i - {l_i}^0\right)^2 \tag{3.3}$$

In order to determine the position of the stage based on the length of the spring, one must know the A_i and B_i coordinates. Because A_i is considered the ground, the coordinates *A_i* are always known. If the *B_i* coordinates are known, one can determine the length of the spring using the following relationship:

$$l_i{}^2 = (B_i - A_i)^T (B_i - A_i) \tag{3.4}$$

Taking the derivative of the potential energy of the springs for the principle of virtual work solution yields the following equations:

$$F_x = \sum_{i=1}^{3} k_i \left(l_i - l_i{}^0 \right) \frac{\partial l_i}{\partial x} \tag{3.5}$$

$$F_y = \sum_{i=1}^{3} k_i \left(l_i - l_i{}^0 \right) \frac{\partial l_i}{\partial y} \tag{3.6}$$

$$T_z = \sum_{i=1}^{3} k_i \left(l_i - l_i{}^0 \right) \frac{\partial l_i}{\partial \theta} \tag{3.7}$$

The change in spring length with respect to *x*, *y*, and Θ is given as

$$\frac{\partial l_i}{\partial x} = (B_i - A_i)^T (0,1) / l_i \tag{3.8}$$

$$\frac{\partial l_i}{\partial y} = (B_i - A_i)^T (1,0) / l_i \tag{3.9}$$

$$\frac{\partial l_i}{\partial \theta} = (B_i - A_i)^T [R'] b_i / l_i \tag{3.10}$$

From the equations (3.5) – (3.10), it is clear that the only tools necessary to determine the forces and torques applied to the system are the Coordinates *A* and *B*, as well as the spring rate and the free length of the spring. If the spring length is known, coordinates B_i can be determined.

3.2 Rigid body solution

In order to simplify the above equations, a rigid body approach was determined to be more suitable. The system was analyzed as a rigid body due to the stiffness of the system with respect to the forces applied. By doing so, the system can be assumed to be rigid, and thus the system reduces to a simple force balance where the sensor values can be read, and converted to force. The main difference between this model and the principle of virtual work model, is that there are no springs in the model. Because of this, the position B_i does not need to be determined.

The first step in determining the rigid body analysis is to determine the system constraints. The outer ring of the transducer (not shown in the figure), as in the principle of virtual work solution, is considered to be the ground by convention. The inner stage, likewise is considered to be constrained to the ground by rigid links that only transfer force in the direction applied. In-plane forces only are applied to the center of the stage in order to isolate the force as a force only and not induce a couple on the system. If forces were applied in a plane offset, a torque would be applied, thus affecting the output. The couple is also applied to the center of the stage. Refer to Figure 11.

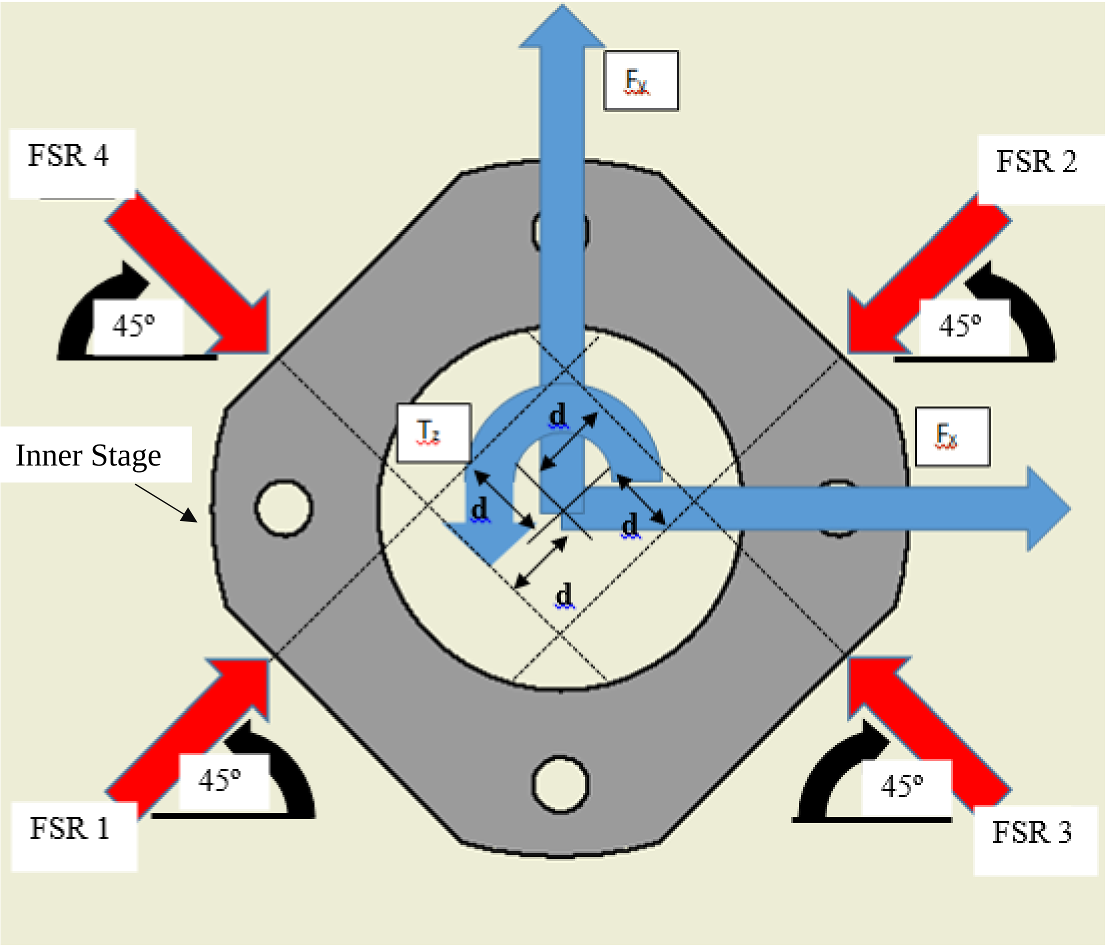


Figure 11: Rigid body model of the prototype transducer

From statics, the equations associated with the model are simply the force and moment equilibrium equations, which are.

$$\sum F_x = 0 \tag{3.11}$$

$$\sum F_y = 0 \tag{3.12}$$

$$\sum T_z = 0 \tag{3.13}$$

For forces in the X-direction, each of the sensors are measured and the x vector component is isolated by multiplying the Sensor by the cosine of the angle it is rotated about the z axis, starting from the x axis. The force balance equation expands to the following equation

$$\sum F_x = f_x + \mathrm{S}_1 \cos 45 - S_2 \cos 45 - S_3 \cos 45 + S_4 \cos 45 = 0 \qquad (3.14)$$

Rearranging, solving for f_x, and simplifying the cosines yields the following relationship. If the sensor values are being read, the force in the x direction can be determined.

$$f_x = -\frac{\sqrt{2}}{2} S_1 + \frac{\sqrt{2}}{2} S_2 + \frac{\sqrt{2}}{2} S_3 - \frac{\sqrt{2}}{2} S_4 \qquad (3.15)$$

For forces in the Y-Directions, the force balance equation expands similarly to the y direction. However, this time, it is a function of sin.

$$\sum F_y = f_y + S_1 \sin 45 - S_2 \sin 45 + S_3 \sin 45 - S_4 \sin 45 = 0 \qquad (3.16)$$

Rearranging, solving for f_y, and simplifying the sines yields the following relationship. If the sensor values are being read, the force in the y direction can be determined.

$$f_y = -\frac{\sqrt{2}}{2} S_1 + \frac{\sqrt{2}}{2} S_2 - \frac{\sqrt{2}}{2} S_3 + \frac{\sqrt{2}}{2} S_4 \qquad (3.17)$$

Measuring torque in the z direction can be found by taking the sum of all of the moments equal to zero about the center. Each of the sensors is offset from the centerline by a distance *d*. The toque equation expands to the following relation.

$$\sum T_z = t_z - dS_1 - dS_2 + dS_3 + dS_4 = 0 \qquad (3.18)$$

Rearranging the above equation yields the following relationship between torque and sensor readings.

$$t_z = dS_1 + dS_2 - dS_3 - dS_4 \qquad (3.19)$$

In matrix terms, the above equations can be shown as a three by 1 matrix of measured variables, $f_{x,}$ $f_{y,}$ $t_{z;}$ equal to a three by four matrix of coefficients multiplied by a four by one matrix of sensors. From this set of equations, given the four sensor readings, the applied forces and torques can be measured.

$$\begin{bmatrix} f_x \\ f_y \\ t_z \end{bmatrix} = \begin{bmatrix} -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ d & d & -d & -d \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix} \qquad (3.20)$$

3.3 FSR Principles of operations and determining the force equation.

The FSRs used in the study were Tekscan Flexiforce A201force sensors. The sensors chosen were the 445 Newton Sensors due to the fact that they are the highest rated FSRs available in the size needed. Although they are 445 Newton sensors, they can withstand up to 4450 Newtons.

The sensors were conditioned and calibrated according to the documentation provided by Tekscan. The sensors were initially loaded vertically with weights statically loaded using the same pins which were used in the transducer.

As shown in Figure 13 below, the relationship between the force and resistance of the sensor follows an inverse power relationship. However, when plotted as force vs, conductance, as seen in Figure 14, the voltage output is quite linear. The equations below show the equations relating force to resistance, given the inverse power law, and relating force to conductance, given the linear relationship. The force output is given as constant A multiplying by the conductance C to give the force. With this, we get:

$$Force = A * \frac{1}{R} \qquad (3.21)$$

$$Force = A * C \qquad (3.22)$$

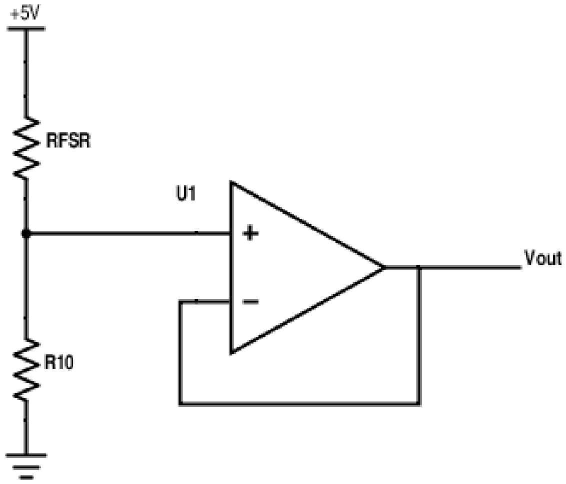


Figure 12: Circuit to process output signal from an FSR sensor

The circuit being used in a voltage dividing circuit shown in Figure 12. Because only voltage is being measured, conductance is found experimentally to be equal to 6750 and is given as the change in voltage, given as

$$C = \frac{1023}{R(1023-V)} \tag{3.23}$$

$$Force = A * \frac{1023}{R(1023-V)} \tag{3.24}$$

Given an applied voltage of 5V, and a 10kohm resistor, as well as the measured Conductance constant A, the force equation reduces to

$$Force = 6750 * \frac{1023}{10(1023-V)} \tag{3.25}$$

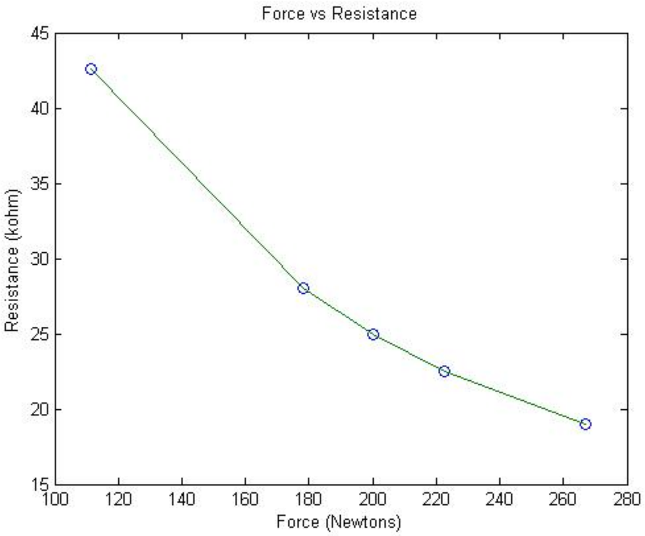


Figure 13: Experimentally determined FSR force vs. resistance curve

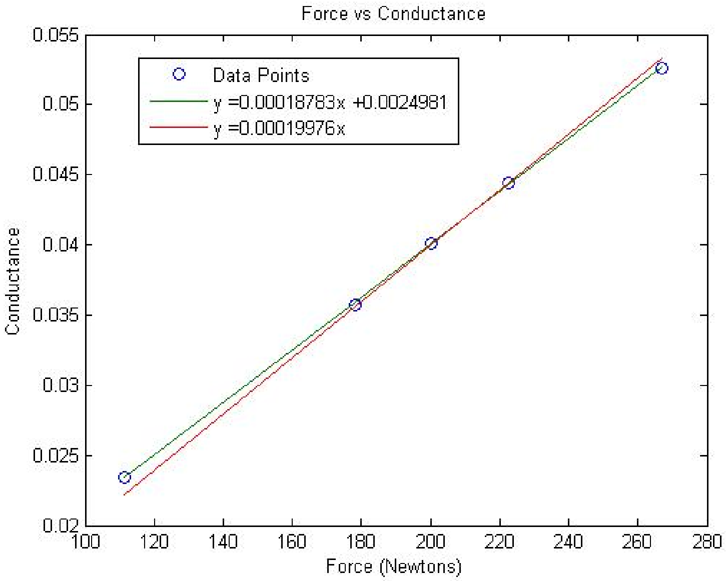


Figure 14: Experimentally determined FSR force vs. conductance curve

3.4 Mechanical System Design

The force and torque transducer was carefully designed, balancing low cost with accuracy and repeatability to create a device that would be able to compete with the current force and torque transducers available on the market. The methodology consists of a carefully designed mechanical transducer, numerical modelling and simulation, and development of the data acquisition system. These three components must all work together in order to ensure that the mechanical design and equations can be verified by the simulation and through testing. For a specification of +/- 100 Newtons force range and +/- 3000 Newton millimeters torque range, the required dimensions were determined to be approximately a diameter of 100 mm and thickness of 25 mm.

The mechanical system consists of five main components. The main components are the outer ring, inner stage, actuation pins, force sensing resistors, and set screws. Each component plays a vital role in measuring the applied forces and torques. Each of the technical drawings for the parts machined can be found in Appendix C. A schematic of the transducer is shown in Figure 15 and a picture of the actual Transducer is shown in Figure 16.

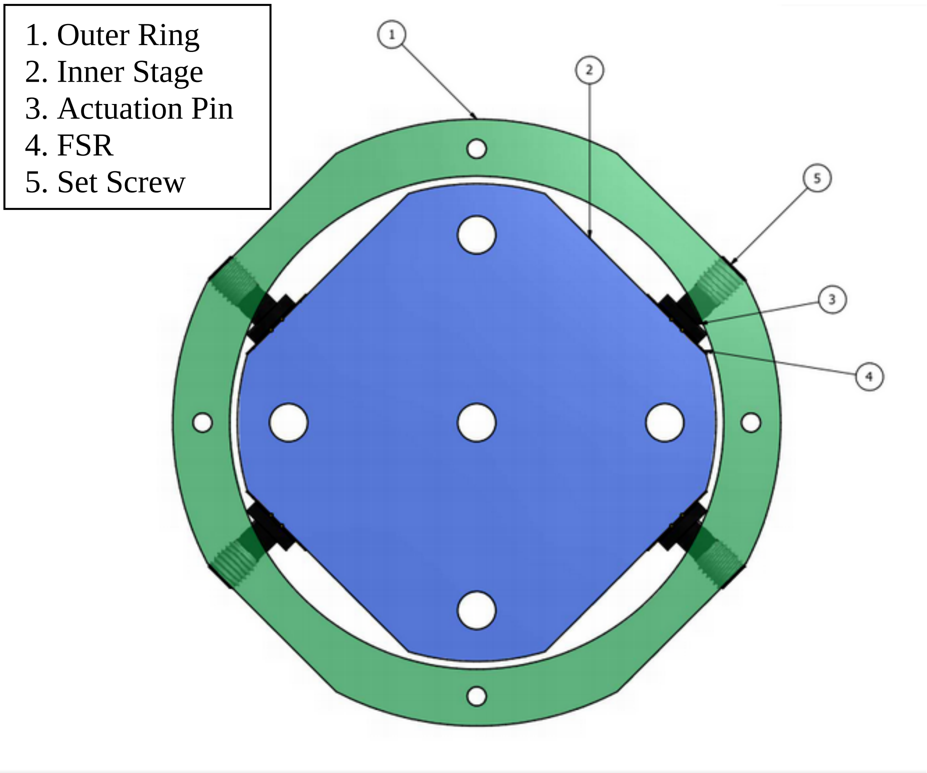


Figure 15: Schematic of prototype transducer

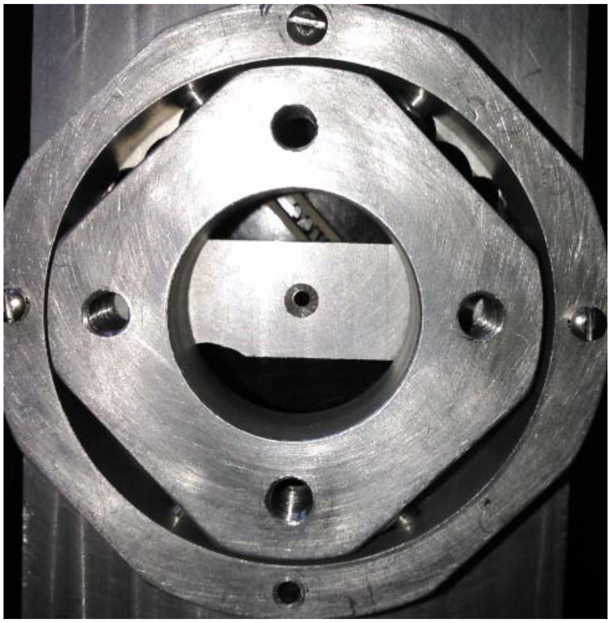


Figure 16: Actual prototype transducer

Each of the machined components are made from 6061 Aluminum, as it is a low cost, widely available material with an excellent balance of strength and weight. While it would be preferable to machine using metric units, the machines used to construct this device measure in terms of imperial units. Therefore, it was determined to create a device using imperial units and convert to the necessary metric units when the actual measurement process is undertaken.

The outer ring is constructed from 4.25” (108 mm) bar stock. The outside diameter of 4” and the inside diameter of 3.25” (82.6 mm) are turned down on a lathe. The thickness between inner and outer diameters of .375” (9.53 mm) was determined to be the minimum thickness allowable in order to ensure the maximum thread count on the set screw while still keeping the thickness to a minimum. The device thickness of .85” (21.6 mm) was chosen due to the size of the sensing element. The four flat edges on the outside of the cylinder are machined by milling the face down. This depth is determined to suit the parameters of the outside diameter as well as a width of 1.2” (30.5 mm). 1.2” was chosen as the minimum width in order to place the set screw .475” (12.1 mm) from the centerline. The holes for the actuation pin and the set screw were chosen to be such that the hole for tapping the quarter inch thread would be the same as the hole for constraining the actuation pin to cut down on unnecessary machining operations.

The dimension of the outer diameter of the inner stage was chosen to be .01” less than the inner diameter of the outer ring. This provides enough tolerance in clearance between the inner stage and the outer ring to ensure that the two are not in contact.

The inner stage also has four flat faces machined to a depth. The depth is chosen based on the diameter of the ring, the offset of the FSR, and the width of the FSR.

The actuation pin is turned down on a lathe from .5” (12.7 mm) 6061 aluminum rod. The schematics of the pin are shown in Figure 18. The smaller diameter is chosen in order to fin inside the hole drilled in the outer ring. The outer diameter is chosen to be .02” (0.05 mm) smaller than the sensing pad of the FSR. This is chosen to apply maximum coverage of the FSR, while still providing enough tolerance to allow for misalignment of the actuation pin.

Proper assembly of the device is imperative in order to ensure the accuracy of the transducer. A thin film of double sided tape is applied directly to the sensing element of the FSR, ensuring that the sensor is flat while being installed. Next, the larger diameter of the actuation pin is applied to the other side of the double sided tape, ensuring that the pin is directly in the center. This step is repeated for all of the remaining actuation pins and FSRs. The inner diameter of the pin is then inserted in the outer ring, with the FSRs all facing the center of the ring. The stage is placed in the center of the assembly, with the flat faces parallel to the FSRs. The last step is to thread in the set screws which will apply the appropriate preload to the force sensing resistors.

The heart of the device is in the actuation system. The actuation system can be seen in Figure 17. The actuation system transmits the applied force through the inner stage, into the FSR. The opposite side of the FSR is held in place by the actuation pin, which is held concentrically constrained by a hole drilled in the outer ring. The set screw is threaded into the outer ring and provides the reaction force necessary to hold the assembly in place.

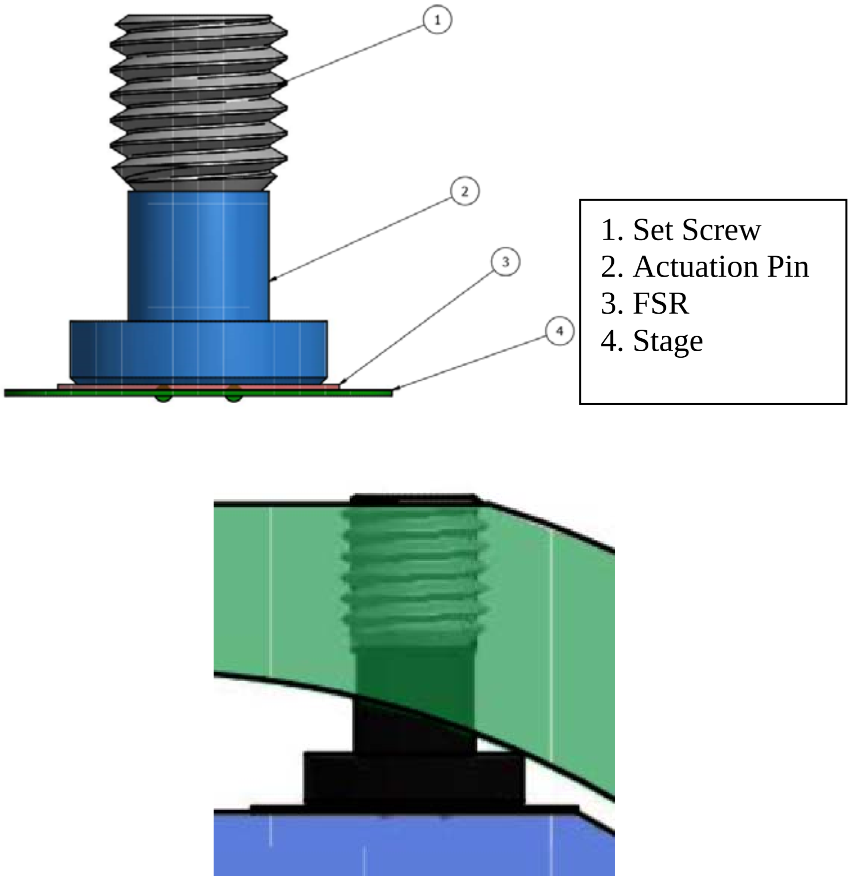


Figure 17. Transducer actuation system

3.5 Simulation

The simulation was run using a combination of three programs, including Matlab/Simulink, Adams/View, and Autodesk Inventor. The actual simulation portion of the testing was carried out using the Adams/View software provided by MSC. The software was an obvious choice for the study as it contained a few key components, including the ability to integrate solid models as well as control and automate the simulation with MATLAB and Simulink. The Simulation software was also an excellent choice over standard FEA software, as it is computationally much faster than running a full FEA simulation for different levels of force.

The simulation set up is very similar to the theoretical Principle of virtual work solution for the planar stage supported by springs. The outer ring of the sensor was considered the ground. The inner stage was supported by four springs, and was not grounded. In order to consider the spring rates, the linear spring constant *k* was determined based on the dimensions of the pin, shown in Figure 18.

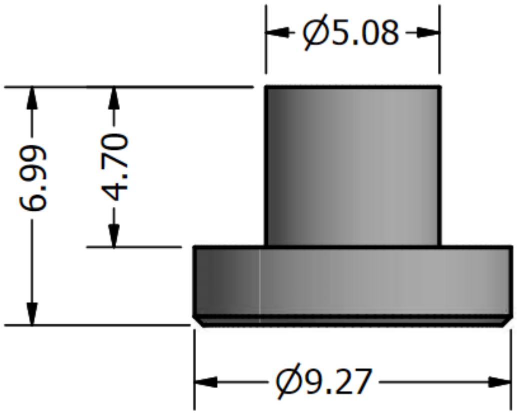


Figure 18: Actuation pin dimensions

The relationship between an axially loaded bar and spring constant *k* are as follows, where the area *A* is the cross sectional area of each segment, the elastic modulus *E* is 69 GPa, and the length *L* of each segment.

$$k = \frac{AE}{L} \tag{3.26}$$

The spring rate for a cylinder with two major diameters can be found to be

$$k_e = \left({k_1}^{-1} + {k_2}^{-1}\right)^{-1} \tag{3.27}$$

By using the above formulas, the spring rate of the pin being used can be determined.

In the actual transducer, the preload will be applied to each individual sensor, equal to half of the full range of the sensor. For the study, the sensor were loaded to 220 Newtons due to the fact that the sensors were rated for 445 Newtons. Each of the values used in the simulation can be found in Table 4.

For each simulation, the magnitude of the forces acting on the spring were recorded. It is important to note that in order to accurately model the FSRs, the force must be a magnitude and not a direction. Recording the forces acting on the spring is equivalent to taking the forces acting on the FSRs. From this data, the forces could be fed into the force-torque equation (equation (3.20)) and the forces and torques acting on the system could be determined. The forces and torques on the system were applied directly to the center of the transducer following the right hand rule.

Table 4: Constants used in Simulation

Spring Constant	k (N/m)	9.25E+10
Pre-Load	P (N)	220
Ground Coordinate (X,Y) (mm)	A1	(-35.4,-35.4)
	A2	(35.4,35.4)
	A3	(35.4,-35.4)
	A4	(-35.4,35.4)
Sensor Coordinate (X,Y) (mm)	B1	(-26,-20)
	B2	(26,20)
	B3	(26,-20)
	B4	(-26,20)

3.6 Integration with Simulink

A key component of the Adams/View software is the fact that it is set up to be used directly with Simulink. Within the system, a series of design variables are created. Following a simple script in Matlab, a series of input and output variables were fed into a plant model. The Simulink model is shown in Figure 20 and Figure 21.

The input variables, labeled Force_x, Force_y, and Torque_z were created and were linked directly to the corresponding forces and torque applied to the simulation. The output variables, which are the measured variables, are the magnitude of the forces acting on each individual spring. The variables are defined as Sensor_1, Sensor_2, Sensor_3, Sensor_4. For each simulation, these forces are measured and sent to Simulink, where they can be stored and plotted.

Using Simulink with Adams View proved to be an invaluable tool in speeding up the simulation process. With Simulink, theoretical applied forces are defined, sent through the plant representing the transducer, the forces on each spring could be recorded, and the overall forces could be determined.

Each simulation required approximately 6 seconds to initiate, run, analyze, and save the data. While this is very fast, when considering large varying forces applied to the transducer, the need to streamline and choose specific simulations becomes quickly apparent. The equation below shows the time required to compute based on different quantities of force and torque simulations

$$Number\ of\ Simulations = (Force)(Angle)(Torque) \tag{3.27}$$

$$Simulation\ Time = 6\,\frac{seconds}{simulation} * Number\ of\ Simulations \tag{3.28}$$

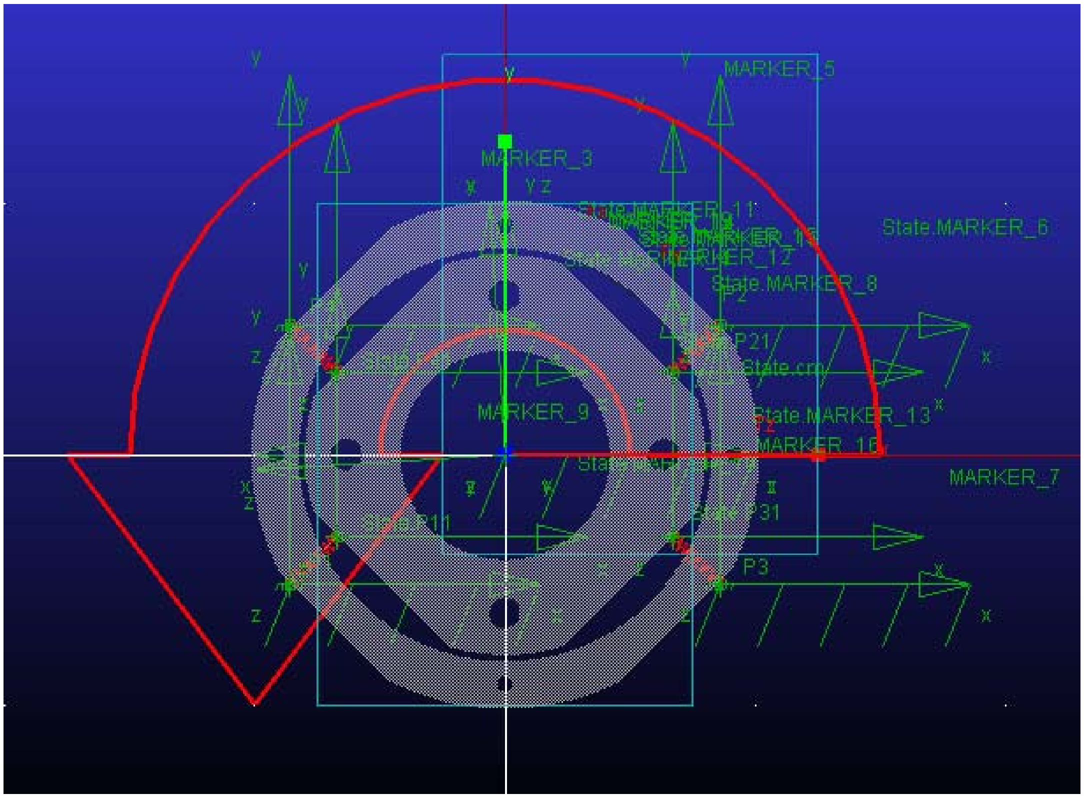


Figure 19: Adams/View simulation of prototype transducer

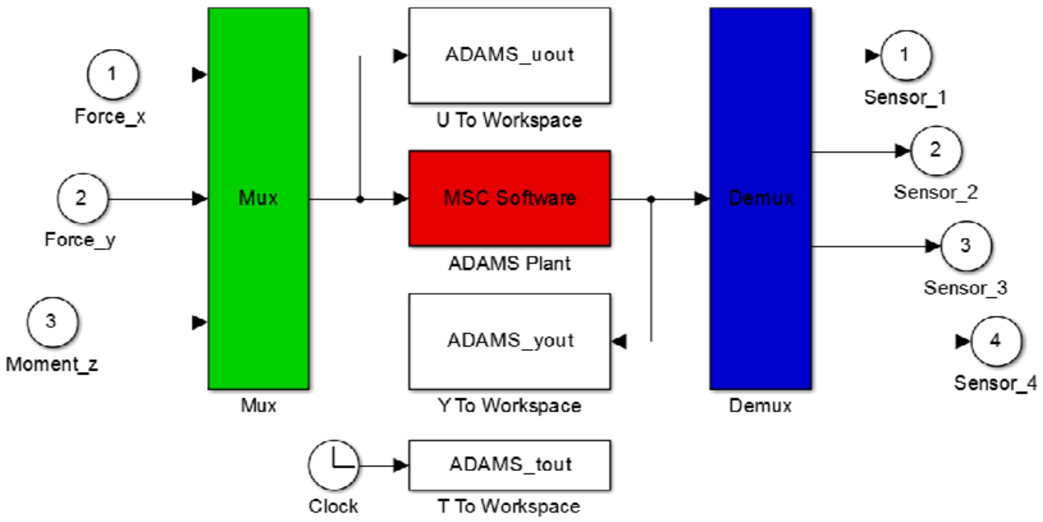


Figure 20: Simulink Adams_sub block

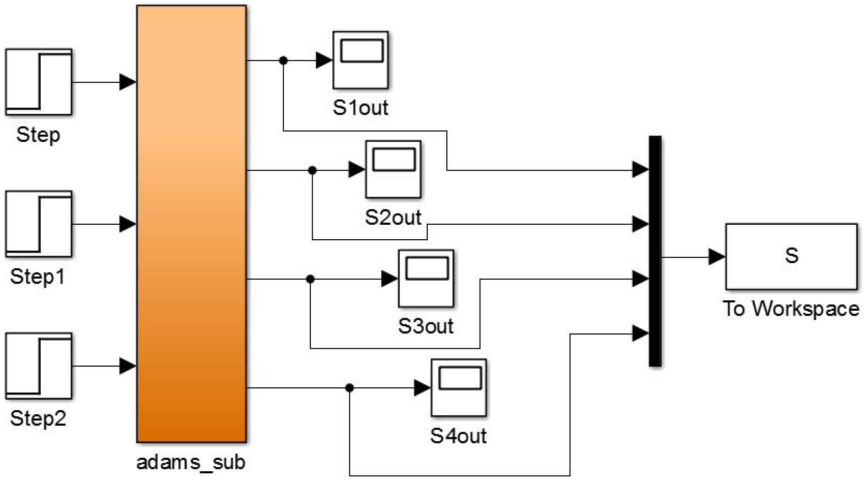


Figure 21: Overall simulink model

3.7 Validity of Rigid Body analysis

In theory, the principle of virtual work is a viable method of solving for the forces and torques associated with the system. However, in practice, it is computationally much faster to use the rigid body model largely due to the required coordinates of the origin as well as any errors in measurement. In order to use the principle of virtual work, the sensor values must be measured and related back to the spring length *l*, given as the following equation, where the measured force *f* is divided by the spring constant *k*.

$$l = \frac{f}{k} \tag{3.20}$$

While this is theoretically easy to accomplish, any errors or inconsistencies in the measured forces of the sensors has the potential to create a combination of spring lengths that is theoretically impossible to achieve, and thus no solution exists. One way to deal with this problem is to use a minimization function to determine a best fit of the spring lengths. These lengths can be used to determine the center of the stage and rotation, whereas the actual position of the stage can be determined and the rest of the principle of virtual work solution can be worked out. While this is feasible, and in practice provides an accurate solution, it is not practical due to the length of time required to solve this equation. In simulation, a maximum calculation speed of 20Hz was achieved, hardly practical for real time measuring.

Using the Adams/View Software, MatLab, and Simulink, a simulation was set up with a rigid, massless stage connected to a rigid grounded outer ring by four springs of spring rate *k*. The spring rate was determined in conjunction with the mechanical design. The plant model of the simulation was exported to Simulink, as described in section 3.6.

A series of pre-determined force and torque combinations were determined. The combinations were saved to a matrix and sent as the inputs into the Adams Plant. The outputs of the Adams plant were the force readings on the springs, which were equal to the spring constants multiplied by the change in length of the springs. The outputs were saved and fed back into equation 3.19 to determine what the sensor output would be for the applied force. This was an important simulation to run, as it correlated the force and torque output of the transducer to the force and torque applied to the transducer.

The results of running the simulation and the results of the rigid body analysis are shown in

Figure 22: through Figure 26. All sets of data show the same results, however, due to the fact that there are three variables. Multiple plots were required to show all results of the simulations. In

Figure 22, F_y and T_z are plotted for multiple levels of Tz. The values of Tz are -1600, -800, 0, 800, and 1600 Nmm. In Figure 23 to Figure 26, F_x and F_y are plotted, while T_z is shown as a color. It is shown that the Rigid Body Model is within a 2% error of the Principle of Virtual Work. This is accurate enough for the transducer being designed.

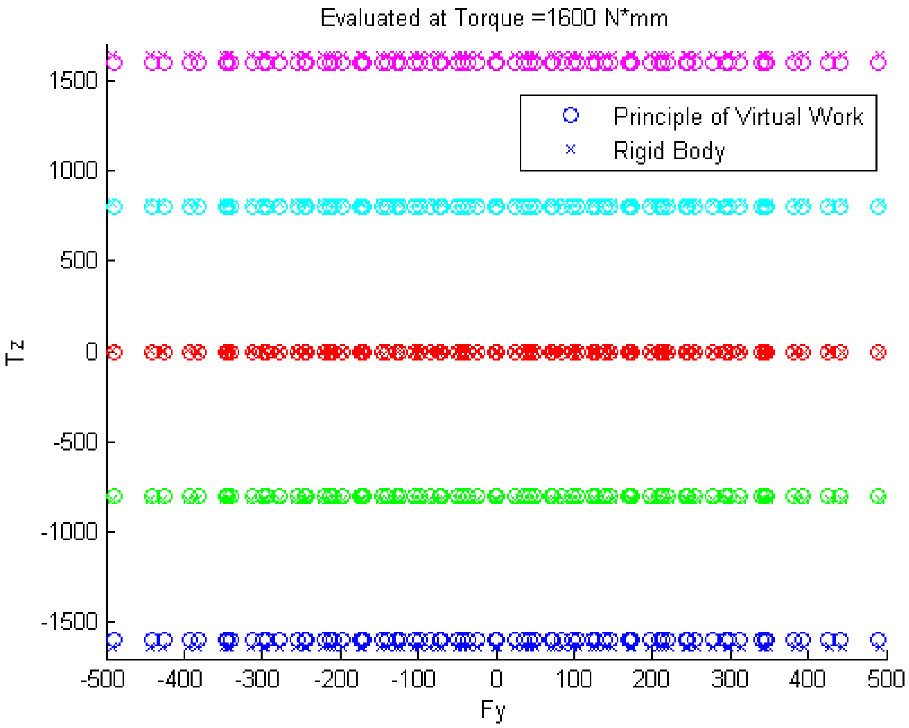


Figure 22: Simulations results plotted as a function of Tz versus Fy

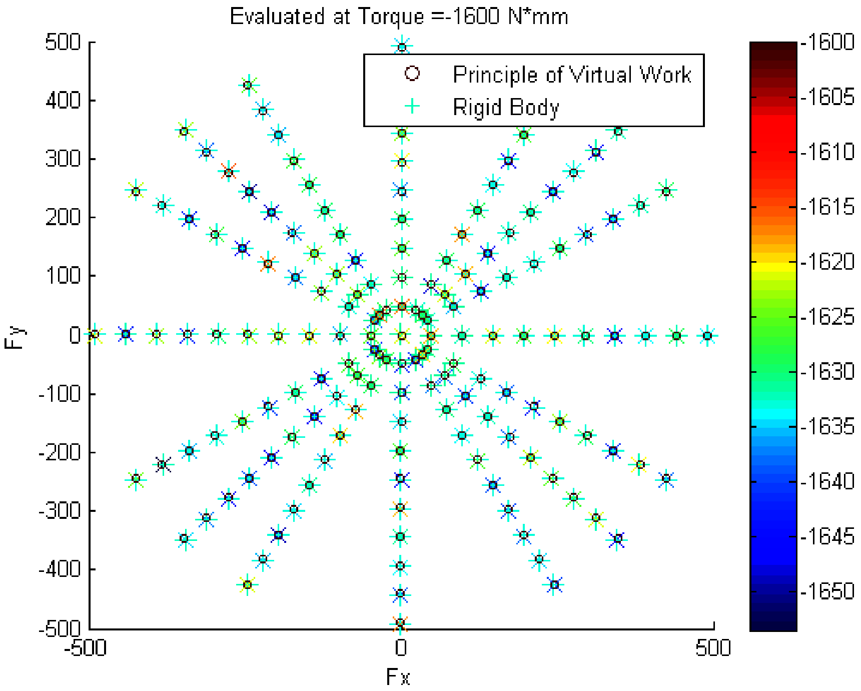


Figure 23: -1600 N mm simulation results plotted as Fy vs Fx

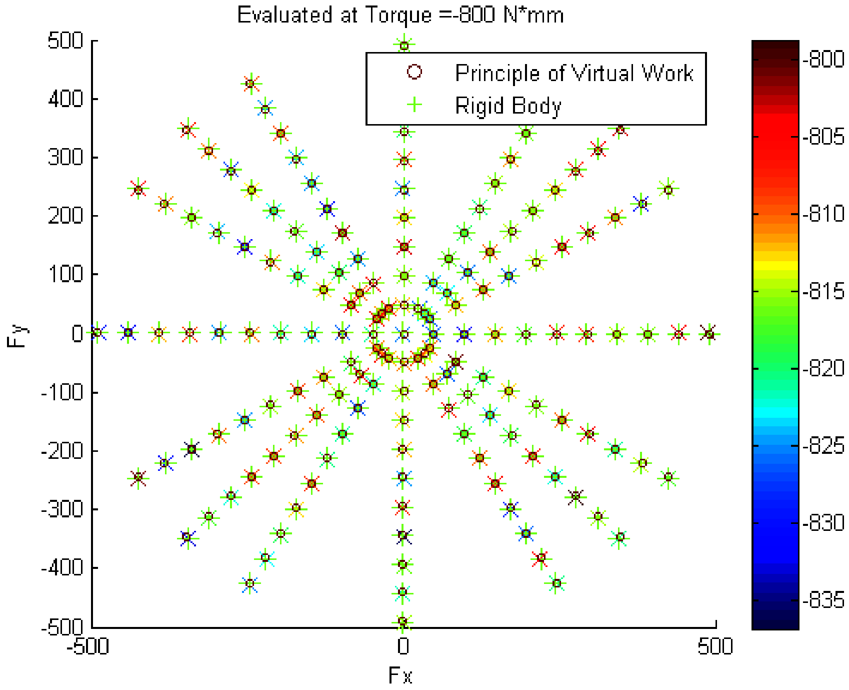


Figure 24: -800 N mm simulation results plotted as Fy vs Fx

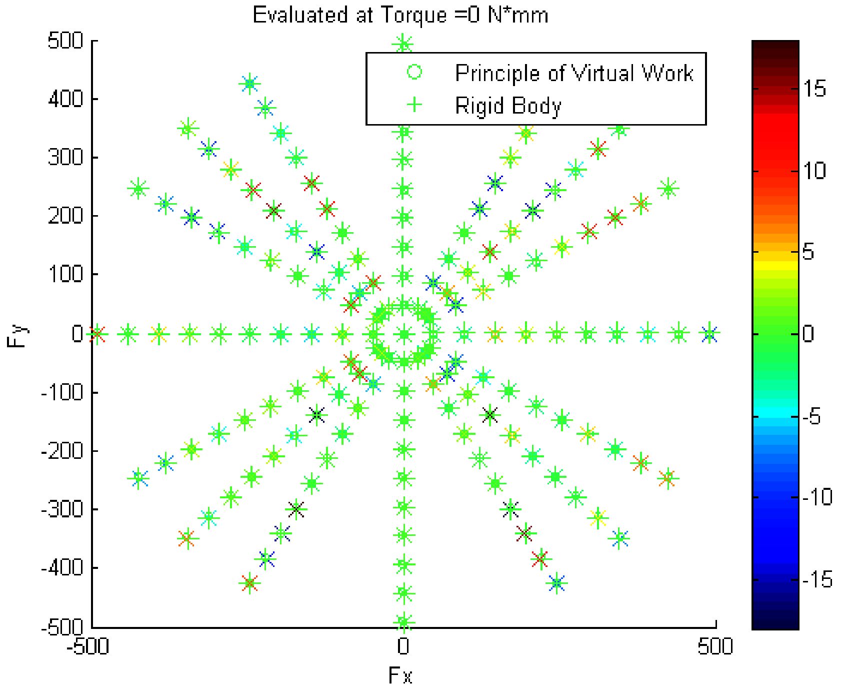


Figure 25: 0 N mm simulation results plotted as Fy vs Fx

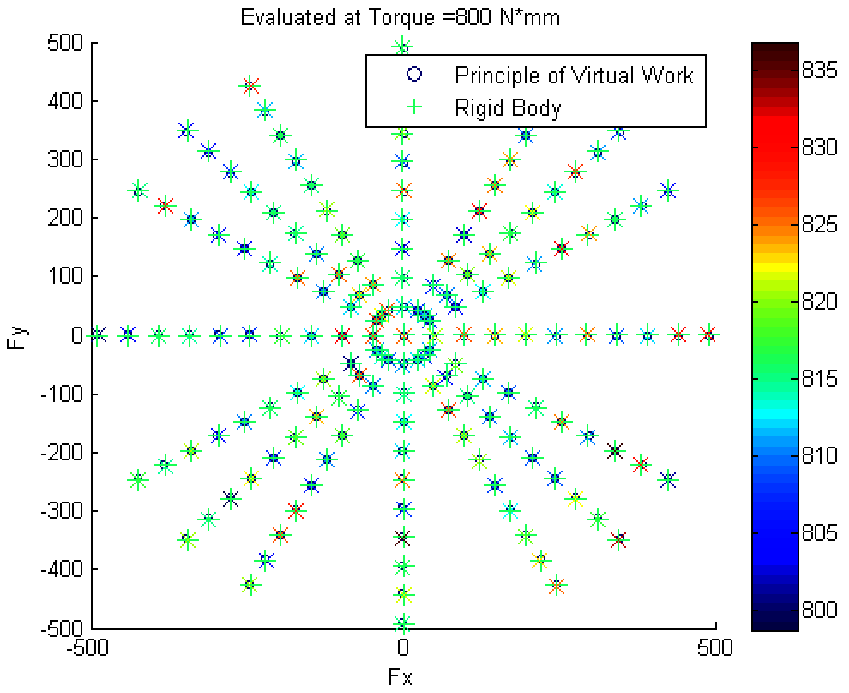


Figure 26: 800 N mm simulation results plotted as Fy vs Fx

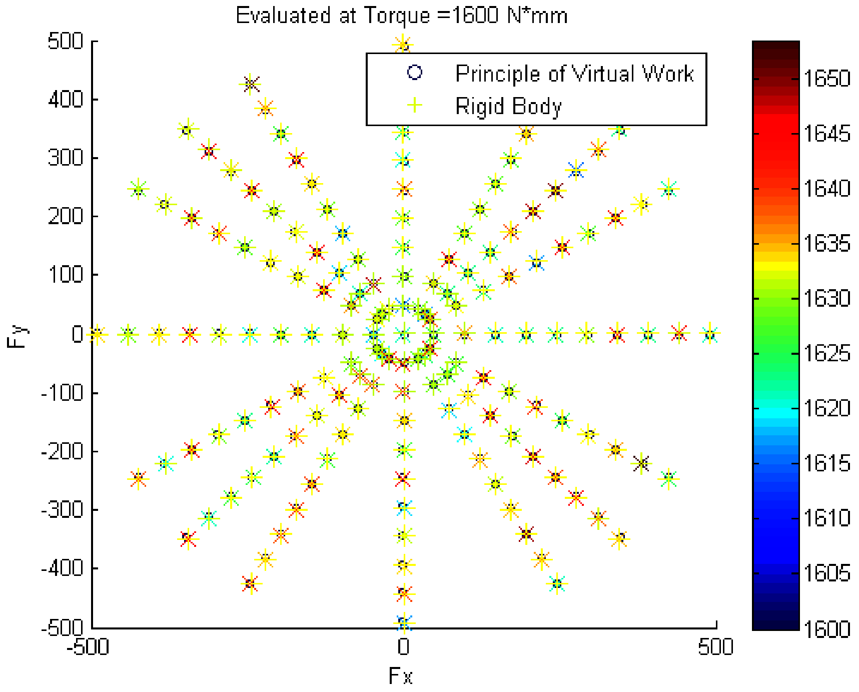


Figure 27: 1600 N mm simulation results plotted as Fy vs Fx

3.8 Data Acquisition

The data acquisition system used for the project consisted of three main aspects, including acquiring the data, converting the raw data to force and torque outputs, and finally saving and analyzing the data.

The Arduino Uno was chosen due to its extremely low cost, ease of use, and ability to accurately acquire data. The Arduino served two main functions. The first of which was to power the circuit for the FSRs, shown previously in Figure 12. The circuit diagram for the entire circuit is shown in Figure 28 . The second function was to acquire data from the FSRs. The code used in the Arduino was quite simple and can be found in Append B. In order to acquire and transmit the data, the Arduino first received the number of samples to be acquired from the user input to Matlab. This number was sent from Matlab to the Arduino via the serial port. A baud rate of 56700 was experimentally chosen as it balanced speed and ability to process data. Once the number of samples to send was determined, the Arduino began to read the Analog to digital ports that were reading the voltages from the circuit. Once read, the Arduino would transmit the values of each FSR. When 5 milliseconds passed, the Arduino would take another reading, until all samples were taken.

Matlab took all of the values transmitted and plugged them into the force torque matrix and determined the forces and torques acting on the system.

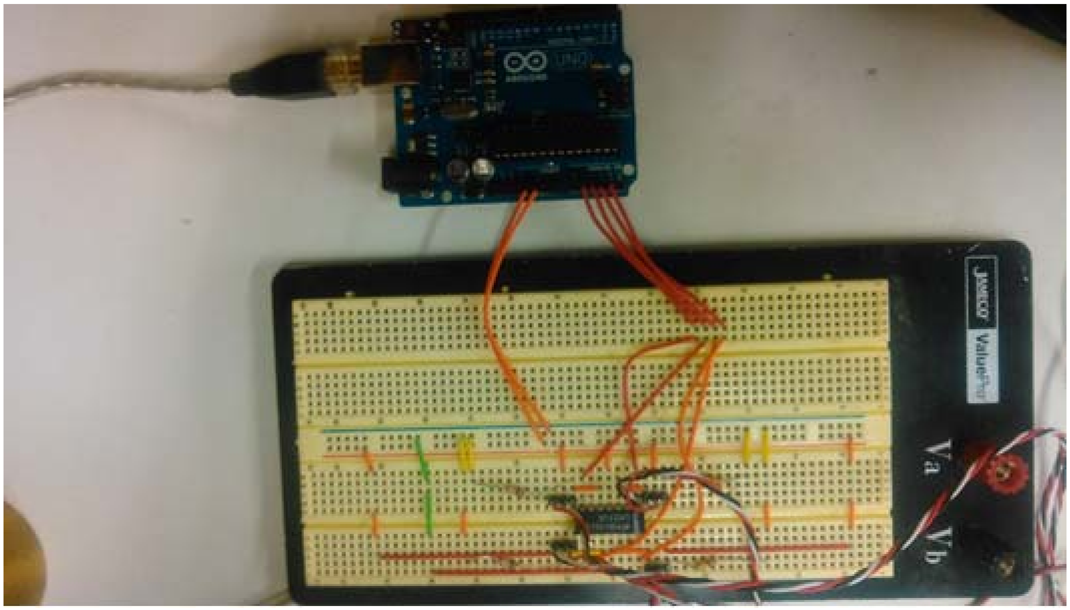
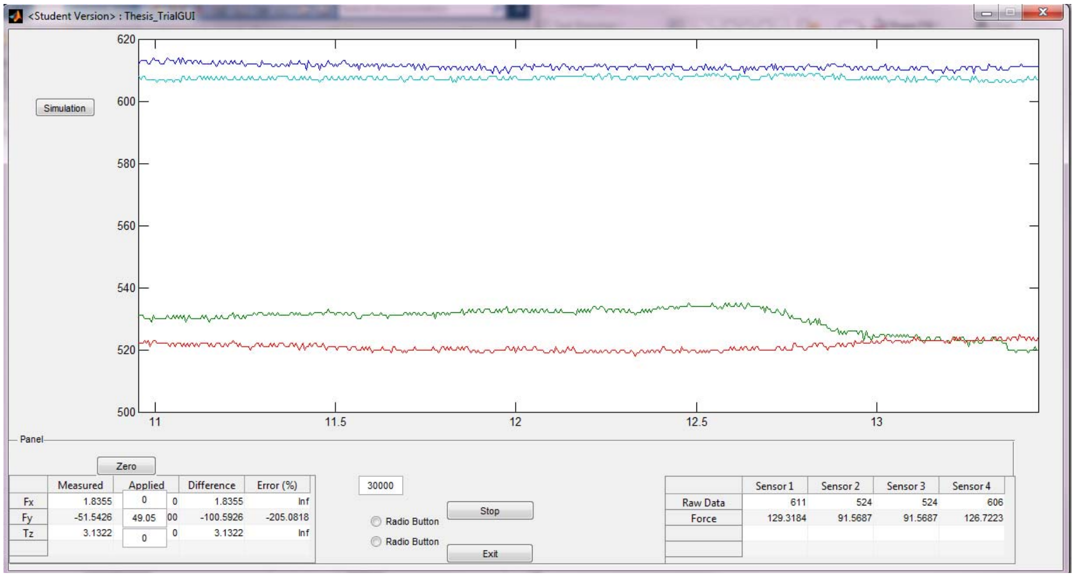


Figure 28: Four circuits (Figure 12) used for FSR data acquisition utilizing

3.9 Overall System

Matlab’s guide GUI application was an obvious choice for easily acquiring data. Matlab has the ability to send information over the serial line as well as receive data, making it very easy to interface with the Arduino. The GUI, or graphical user interface, is shown in Figure 29. Further information on the code can be found in Appendix A.

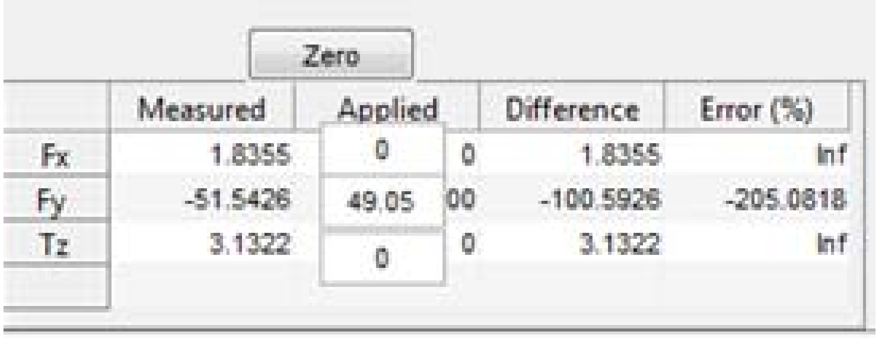


	Measured	Applied	Difference	Error (%)
Fx	1.8355	0 0	1.8355	Inf
Fy	-51.5426	49.05 00	-100.5926	-205.0818
Tz	3.1322	0 0	3.1322	Inf

	Sensor 1	Sensor 2	Sensor 3	Sensor 4
Raw Data	611	524	524	606
Force	129.3184	91.5687	91.5687	126.7223

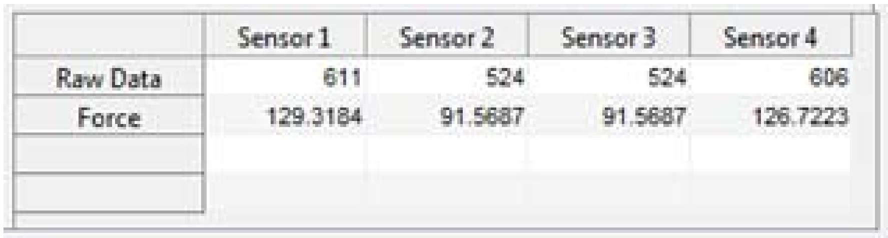
Figure 29: Matlab GUI for testing and data acquisition

The graphical user interface includes a few key features, including the ability to start and stop the test, save the test to a specific location, open all plots associated with the previous test, measure the error in force reading and applied force, measure the sensor data, run a simulated tests, and plot in real time the transducer force and torque values as well as the sensor force values. A close up of the Transducer force display and the FSR force display are shown in Figure 30 and Figure 31.



	Measured	Applied	Difference	Error (%)
Fx	1.8355	0 0	1.8355	Inf
Fy	-51.5426	49.05 00	-100.5926	-205.0818
Tz	3.1322	0 0	3.1322	Inf

Figure 30: Force display in Matlab GUI



	Sensor 1	Sensor 2	Sensor 3	Sensor 4
Raw Data	611	524	524	606
Force	129.3184	91.5687	91.5687	126.7223

Figure 31: FSR force display in Matlab GUI

CHAPTER 4

FINDINGS

The main purpose of the novel force-torque sensor is to be able to sense applied planar forces and torques. The design analysis through the use of both a rigid body model and a dynamic model prove the theoretical capabilities of the device. The next step in determining the feasibility of using FSRs in a force torque transducer is to actually test the performance of the device and relate it back to theory.

The experiments outlined in the following section serve to outline the most important aspects of the device. Testing procedure was carried out in a logical timeline in order to test the most elementary aspect of the device first and progressively add complexity and variability to the device to attempt to simulate real world usage of the device. The methods of testing carried out, in order, include static symmetric loading, static loading on the face of the sensor, both including unloading in between force levels and not unloading, loading at an angle offset, and combination of force and torque loading. These tests test the key properties of the transducer.

4.1 Experimental Apparatus

The experimental apparatus was designed to accurately isolate the types of loading displayed above. The apparatus is constructed from two pieces of aluminum channel, as shown in Figure 32. The first piece serves as the level base in which the upright piece is attached to. The upright piece of channel has a hole milled in it big enough for the tails of the sensors to be fed through the back. It also has threaded holes in order to attach the transducer to.

In order to test for different values of force and torque, weights were hung from the transducer. The weights available for the testing included 1kg, 2 kg, and 5kg brass weights. The weights, used in combination, allowed for testing from 1kg to 8kg, except for the 4kg combination.

Each test below was carried out a total of 9 times for each direction of the testing method to ensure the accuracy of each test. For Figure 35 to Figure 49, the following conventions are defined in order to analyze the data. The independent axis is the force applied. The force recorded is the variable and is plotted as a function of C on the Y axis. For each force level, the average force recorded is indicated by a black circle, and the range of all forces applied is the vertical black lines. The black dashed line is a linear curve fit determined using the least squares solution. The forces applied can then be divided by this line to normalize the force reading to the force applied and thus the applied force will be read. The other line on the plots is labeled "Theoretical Loading." Because it is theoretical, if 10 Newtons are applied, in the theory, one should expect the transducer to read 10 Newtons. Thus, this curve has a slope of 1 and is the theoretical force that should be measured for the applied load.

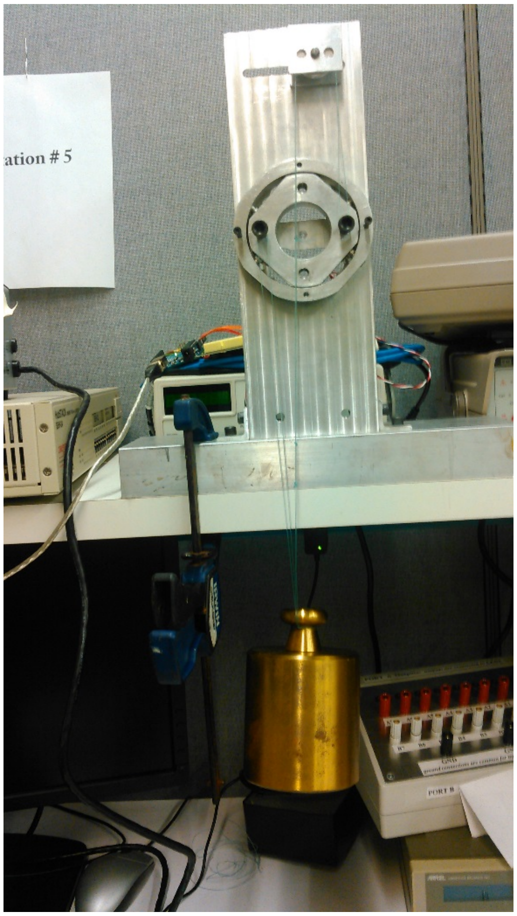


Figure 32: Experimental setup used for symmetric loading

4.2 Symmetric Loading

The symmetric loading test was the most basic test and a clear choice for the first. In order to determine the specific characteristics of the sensor, the sensor was attached to the testing mechanism as shown in Figure 32: Experimental setup. The transducer was loaded in 6 directions, including the X, Y, and rotation in the Z, both positive and negative. Each direction was tested three times per day, for five days, testing every other day for a total of 9 tests (3 day of testing separated by a day of no testing). Each test was performed for a total of 150 seconds. The test was a series of alternating loading and unloading of the device, whereas the device was initially unloaded for ten seconds, then a 1kg mass was applied for ten seconds. The mass was then removed and the 2 kg mass was applied for ten seconds. This procedure was repeated up through 8kg, excluding 4kg due to the lack of availability, finally being unloaded for 10 seconds. A plot is shown in Figure 33 to better visually display the procedure. The purpose of this was to prove two things. The first of which is the repeatability of the test set up. This test proved that the mechanism is repeatable when tested over a period of time. The test also established a base line curve for the device.

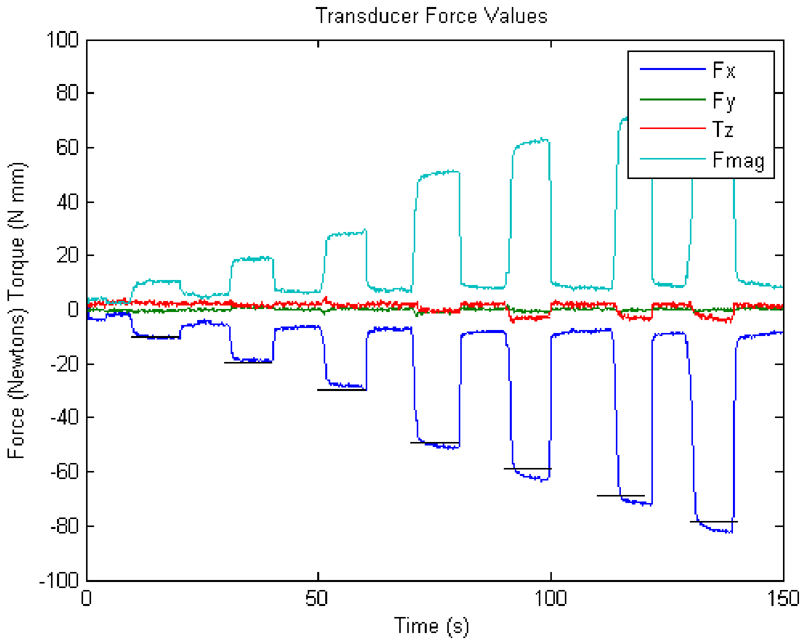


Figure 33: Test results from symmetrically loaded test

The details of the testing are plotted in Figure 35 to Figure 37 and the results are displayed in Table 5. For the first test, the results showed high linearity and correlation with the theoretical results. The results of the testing show two very important characteristics of the transducer. First, the sensor is very linear with respect to the applied force. There is approximately a one percent error between the force applied and the force measured.

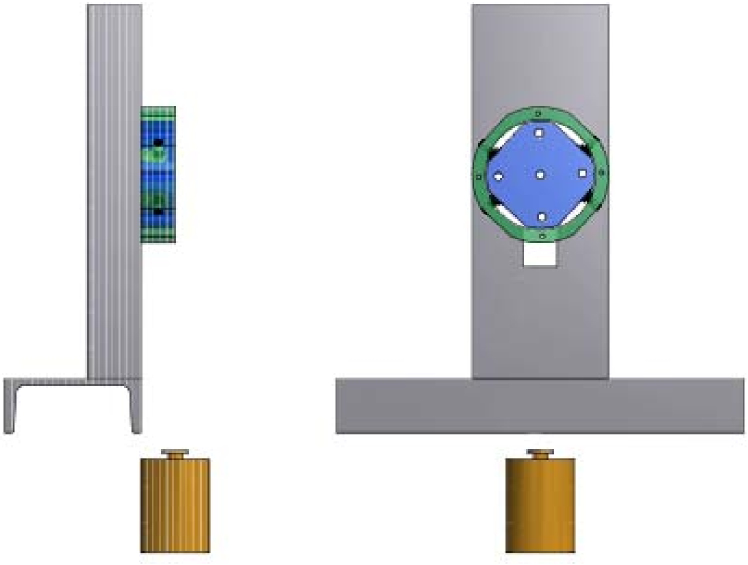


Figure 34: Symmetric experimental set up

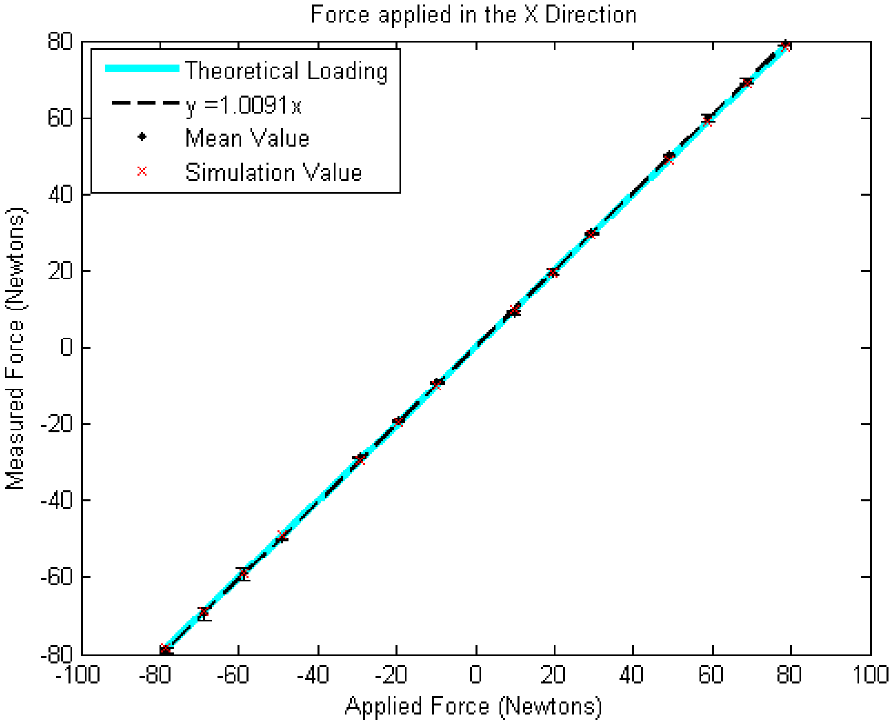


Figure 35: Measured and simulated force for symmetric loading in X direction

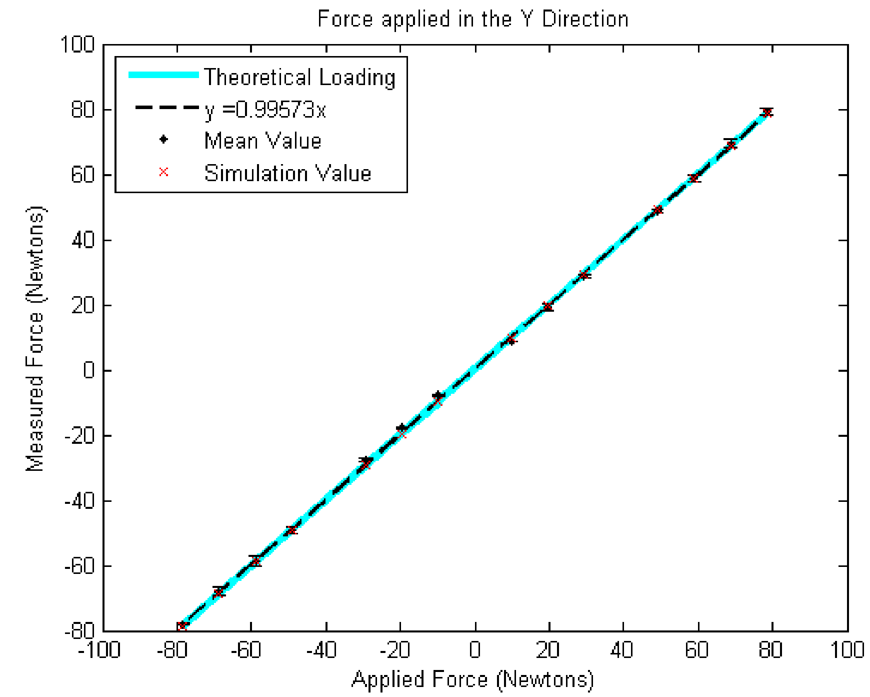


Figure 36: Measured and simulated force for symmetric loading in Y direction

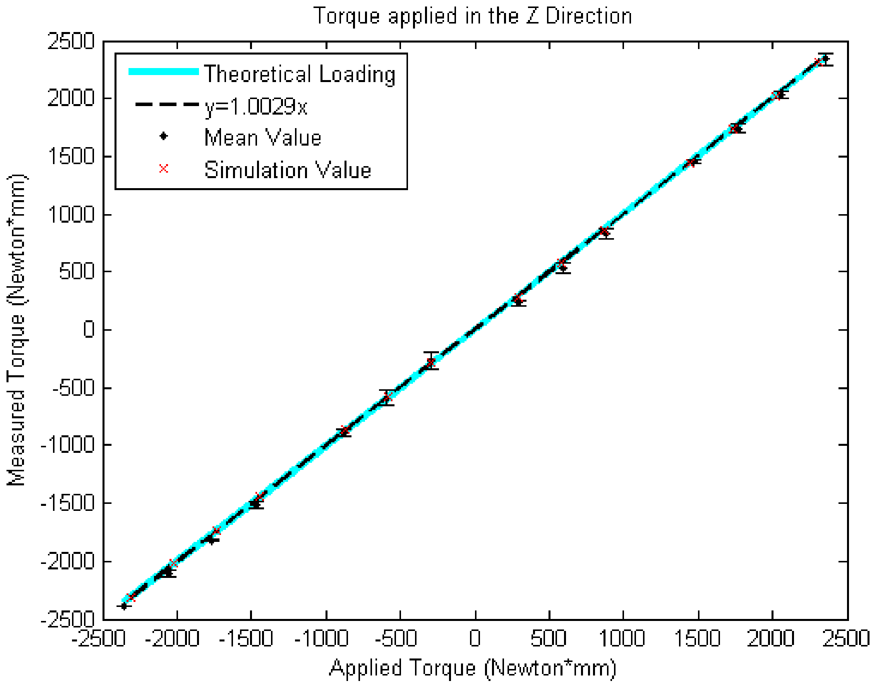
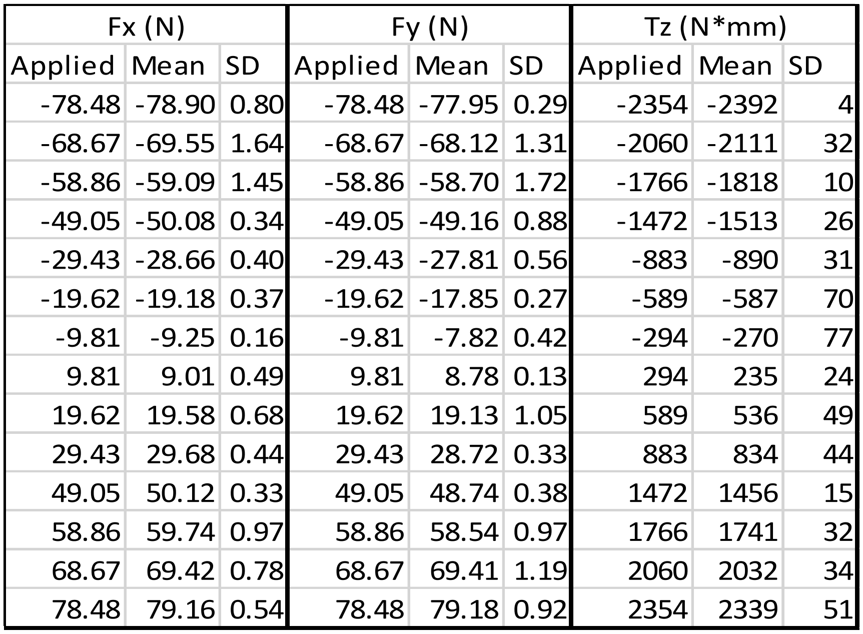


Figure 37: Measured and simulated torque for symmetric loading in Z direction

Table 5: Mean and standard deviation for symmetric loading



Fx (N)			Fy (N)			Tz (N*mm)		
Applied	Mean	SD	Applied	Mean	SD	Applied	Mean	SD
-78.48	-78.90	0.80	-78.48	-77.95	0.29	-2354	-2392	4
-68.67	-69.55	1.64	-68.67	-68.12	1.31	-2060	-2111	32
-58.86	-59.09	1.45	-58.86	-58.70	1.72	-1766	-1818	10
-49.05	-50.08	0.34	-49.05	-49.16	0.88	-1472	-1513	26
-29.43	-28.66	0.40	-29.43	-27.81	0.56	-883	-890	31
-19.62	-19.18	0.37	-19.62	-17.85	0.27	-589	-587	70
-9.81	-9.25	0.16	-9.81	-7.82	0.42	-294	-270	77
9.81	9.01	0.49	9.81	8.78	0.13	294	235	24
19.62	19.58	0.68	19.62	19.13	1.05	589	536	49
29.43	29.68	0.44	29.43	28.72	0.33	883	834	44
49.05	50.12	0.33	49.05	48.74	0.38	1472	1456	15
58.86	59.74	0.97	58.86	58.54	0.97	1766	1741	32
68.67	69.42	0.78	68.67	69.41	1.19	2060	2032	34
78.48	79.16	0.54	78.48	79.18	0.92	2354	2339	51

4.3 Loading the face of the sensor

The second and third tests loaded the forces as shown in Figure 38. The following tests were conducted to show the devices capabilities of measuring forces as they would be applied in the real world. The forces were no longer applied vertically in this test and were not symmetric either. In this test, the sensor was laid down horizontally, as it would in a real world situations. The loading was applied at the face of the sensor. Two tests were initially conducted in this setting. The first was loading the transducer the same as the loading and unloading procedure as in the previous test. In the second method, the transducer was loaded incrementally as before, but without the breaks. Below are the results of both sets of test.

As shown in these test, the results produced a higher level of error than in the first set of tests. The result are shown in Figure 39-

Figure 44: and the mean and standard deviation for each test are shown in Table 6 and Table 7. The data shows that a higher output was produced than in the previous test. This is due to the fact that the loading is no longer symmetric. By loading on the face of the sensor, an additional torque is applied around the x and y axes, and affects the output of each of the sensors. The overall effect is shown in the fact that the output of the device increases by approximately 3–5%. Interestingly, the continuous loading of the sensor created a higher level of error approximately 7–9%. While the exact cause of this is unknown, it shows that the response of the sensor differs based on whether or not it is being loaded progressively or not.

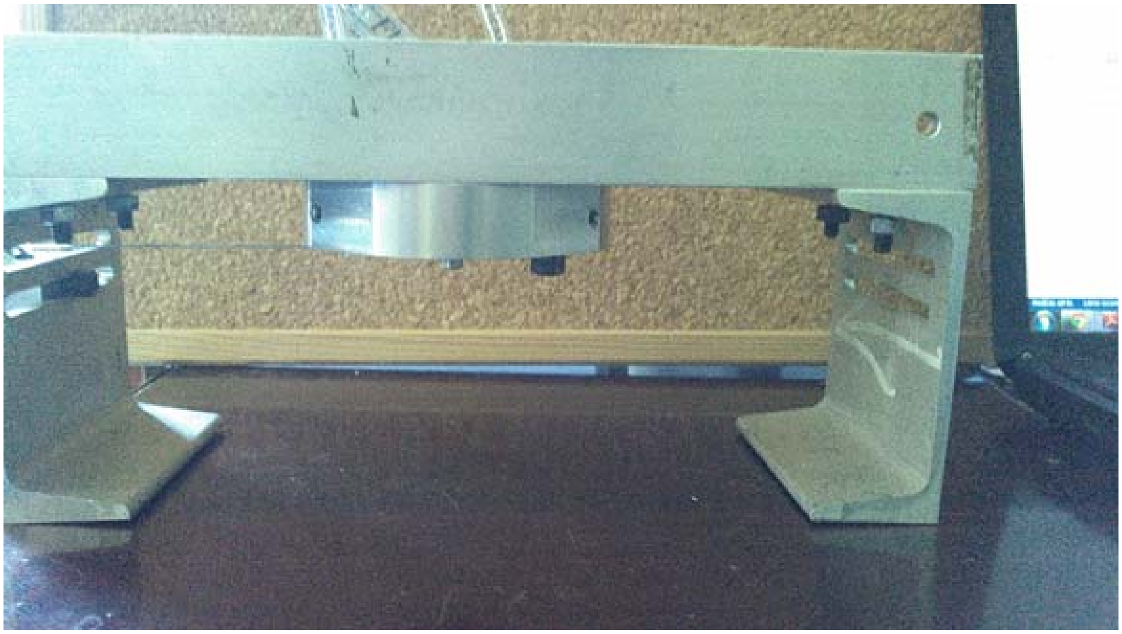


Figure 38: Horizontal loading Set up

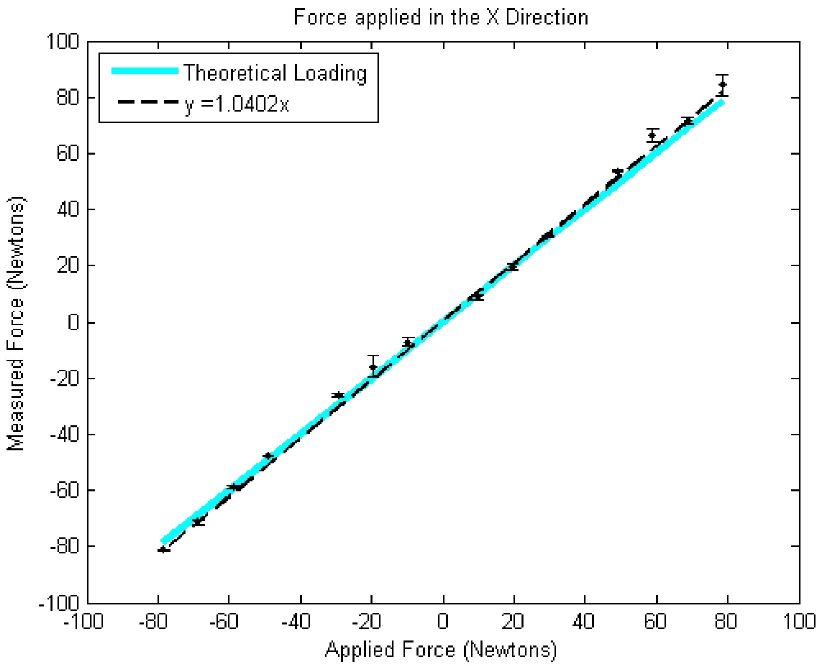


Figure 39: Measured and simulated force for horizontal loading applied to the face of the transducer in X direction

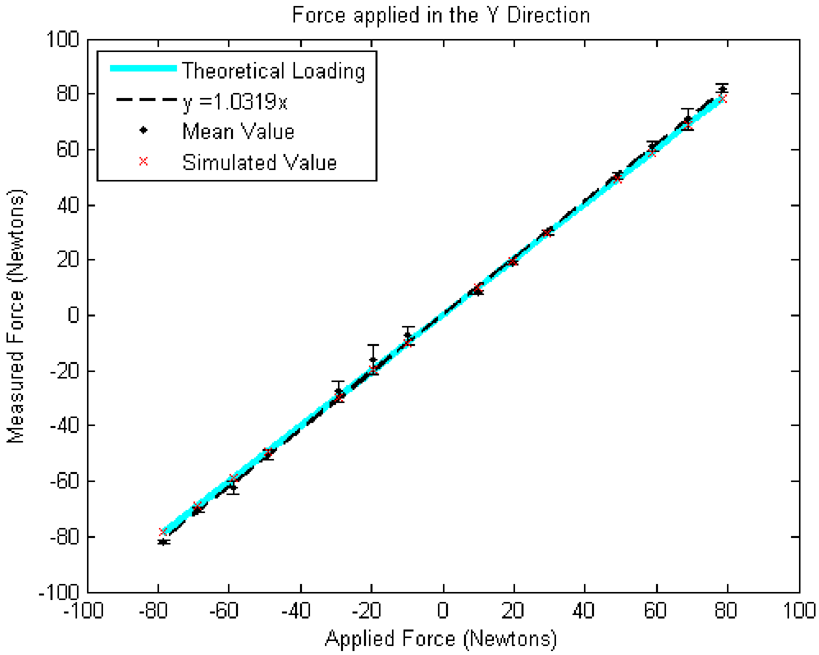


Figure 40: Measured and simulated force for horizontal loading applied to the face of the transducer in Y direction

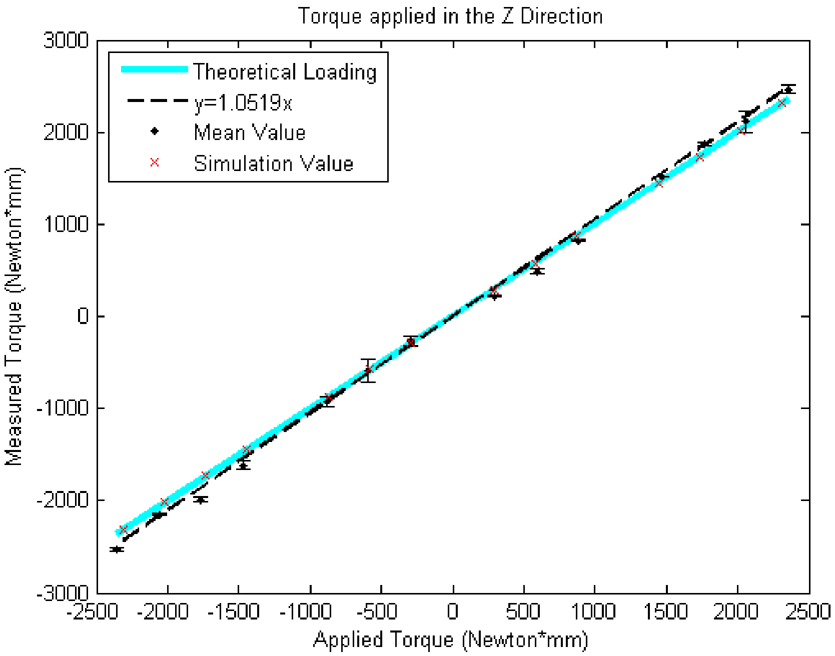


Figure 41: Measured and simulated force for horizontal loading applied to the face of the transducer in Z direction

Table 6: Mean and standard deviation for loading-unloading the face of the transducer

	Fx (N)			Fy (N)			Tz (N*mm)	
Applied	Mean	SD	Applied	Mean	SD	Applied	Mean	SD
-78.48	-81.21	0.38	-78.48	-82.10	0.54	-2354	-2530	18
-68.67	-71.11	0.91	-68.67	-70.35	0.85	-2060	-2147	13
-58.86	-58.71	0.37	-58.86	-62.40	2.37	-1766	-1991	26
-49.05	-47.55	0.09	-49.05	-50.35	1.76	-1472	-1613	39
-29.43	-26.08	0.41	-29.43	-27.44	3.77	-883	-919	55
-19.62	-15.89	3.91	-19.62	-16.28	5.31	-589	-586	118
-9.81	-7.11	1.31	-9.81	-7.35	3.42	-294	-267	51
9.81	8.89	0.72	9.81	8.32	0.59	294	220	12
19.62	19.53	1.09	19.62	19.10	0.42	589	489	27
29.43	30.64	0.36	29.43	29.98	0.88	883	823	11
49.05	53.77	0.23	49.05	50.30	1.30	1472	1510	3
58.86	66.36	2.18	58.86	61.03	1.83	1766	1872	12
68.67	71.58	0.96	68.67	71.07	3.94	2060	2111	117
78.48	84.32	3.95	78.48	82.12	1.69	2354	2463	39

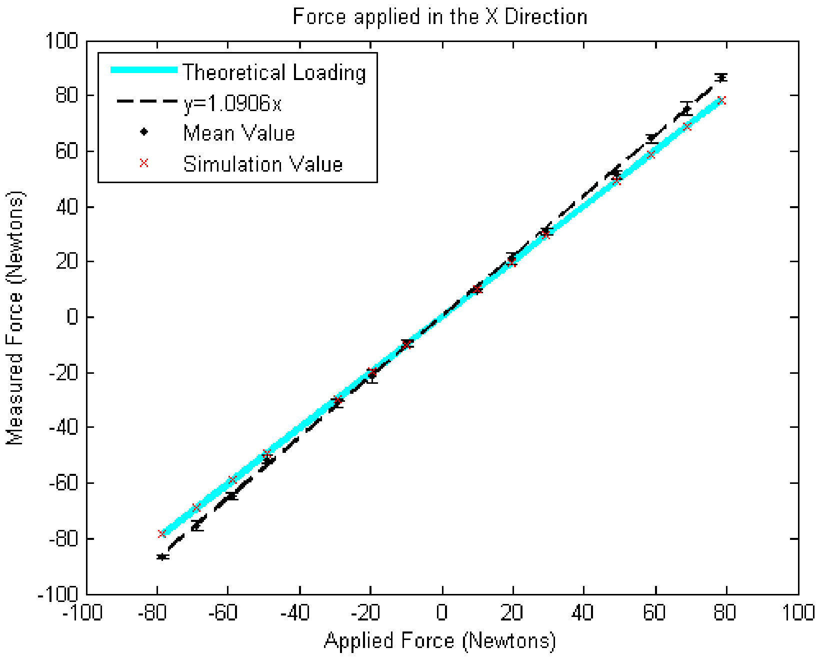


Figure 42: Measured and simulated force for horizontal continuous loading applied to the face of the transducer in X direction

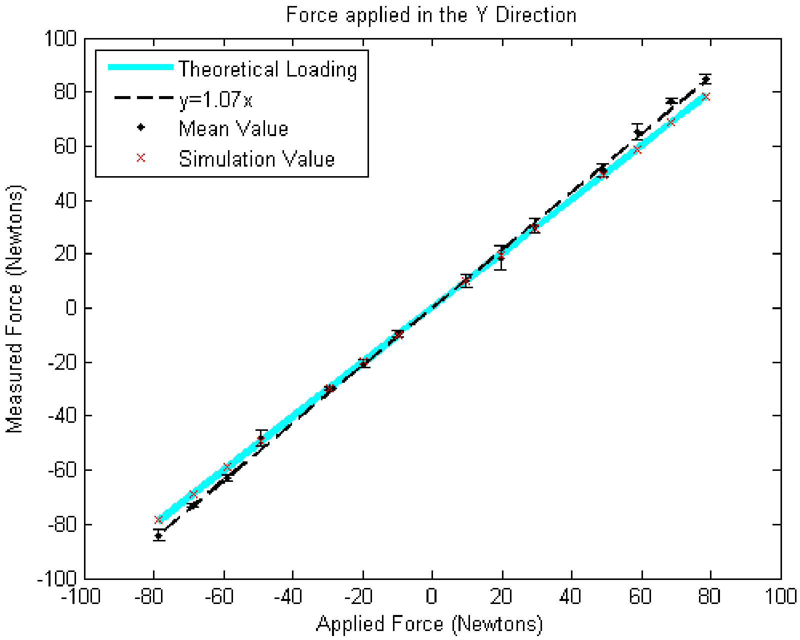


Figure 43: Measured and simulated force for horizontal continuous loading applied to the face of the transducer in Y direction

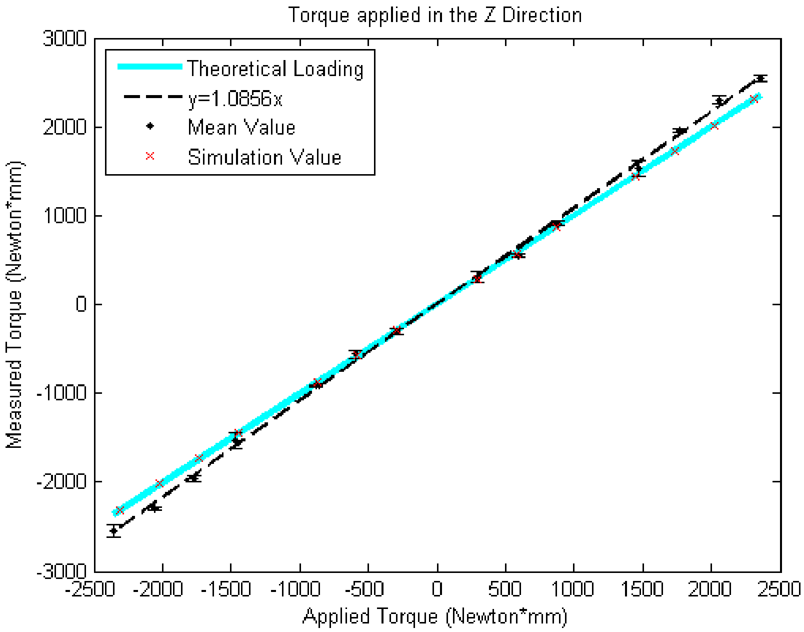
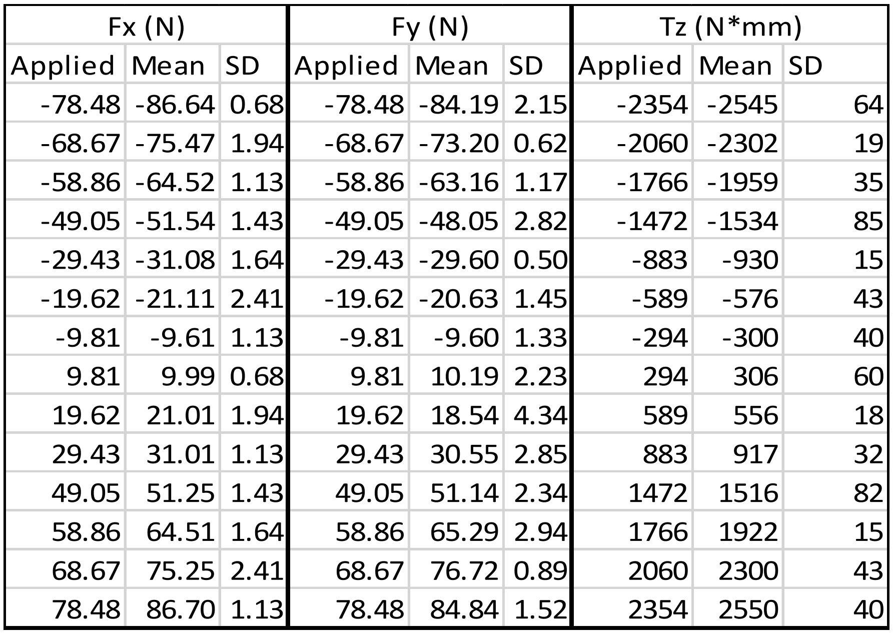


Figure 44: Measured and simulated force for horizontal continuous loading applied to the face of the transducer in Z direction

Table 7: Mean and standard deviation for continous loading the face of thetransducer



Fx (N)			Fy (N)			Tz (N*mm)		
Applied	Mean	SD	Applied	Mean	SD	Applied	Mean	SD
-78.48	-86.64	0.68	-78.48	-84.19	2.15	-2354	-2545	64
-68.67	-75.47	1.94	-68.67	-73.20	0.62	-2060	-2302	19
-58.86	-64.52	1.13	-58.86	-63.16	1.17	-1766	-1959	35
-49.05	-51.54	1.43	-49.05	-48.05	2.82	-1472	-1534	85
-29.43	-31.08	1.64	-29.43	-29.60	0.50	-883	-930	15
-19.62	-21.11	2.41	-19.62	-20.63	1.45	-589	-576	43
-9.81	-9.61	1.13	-9.81	-9.60	1.33	-294	-300	40
9.81	9.99	0.68	9.81	10.19	2.23	294	306	60
19.62	21.01	1.94	19.62	18.54	4.34	589	556	18
29.43	31.01	1.13	29.43	30.55	2.85	883	917	32
49.05	51.25	1.43	49.05	51.14	2.34	1472	1516	82
58.86	64.51	1.64	58.86	65.29	2.94	1766	1922	15
68.67	75.25	2.41	68.67	76.72	0.89	2060	2300	43
78.48	86.70	1.13	78.48	84.84	1.52	2354	2550	40

4.4 Combined Force and Torque

The fourth test that was performed was to load forces and torques at the same time. Given the previous set of preliminary tests where forces and torque were isolated, this test set out to prove that the sensor could in fact measure both forces and torques. This test was carried out with continuous loading, as carried out in the previous test. The results of the test are shown in Figure 45 to Figure 48. The mean and standard deviation are shown in Measured and simulated torque for combined force and torque loading in the X direction

Table 8. This test proves that the repeatability suffers when combining both forces and torques. While this was to be expected given the generally higher level of error for pure torque loading than for pure force loading, the exact cause of this is not known. A few explanations for the cause of this error have been speculated. The fact

that the force is applied at an offset, as stated previously, is a contributing factor to the error. However, what adds to the error is the fact that the device is now being applied at an offset from the center. Because of this, it is causing the device to twist out of plane. The last cause of error is the low number of tests run. Had more samples been run, the overall error would be expected to be lower. These reasons are speculated to be the reason for why the error was so high. However, the device is still within a reasonable level of accuracy given then accuracy of the sensors.

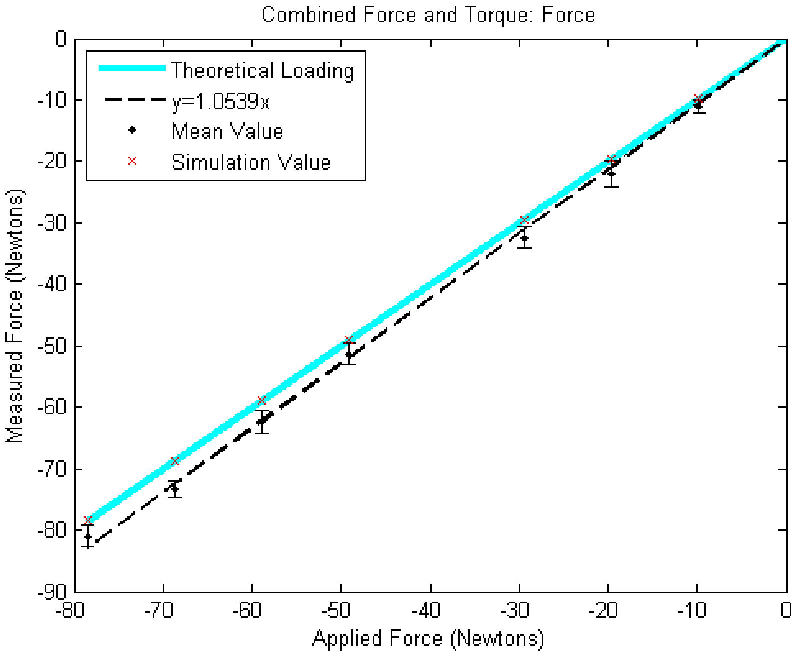


Figure 45: Measured and simulated force for combined force and torque loading in the Y direction

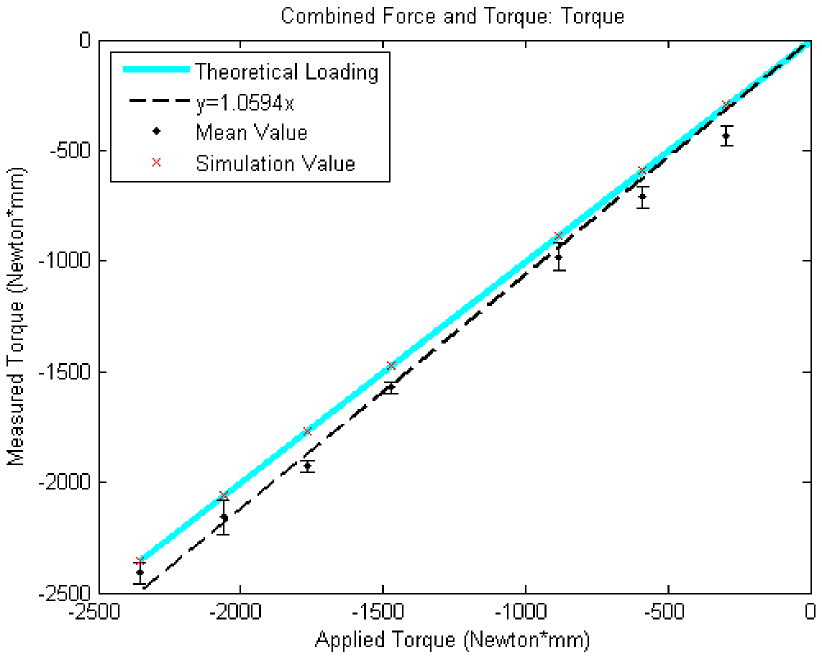


Figure 46: Measured and simulated torque for combined force and torque loading in the Y direction

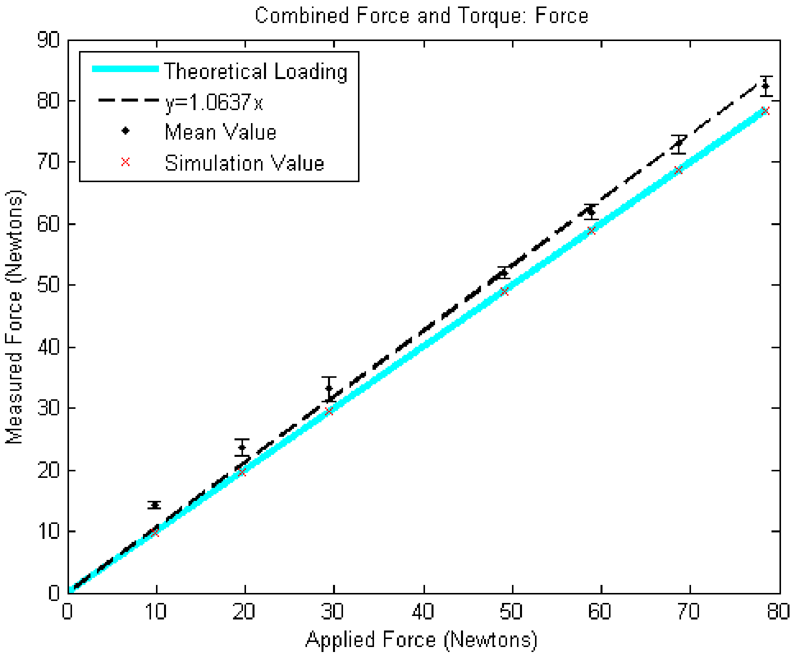


Figure 47: Measured and simulated force for combined force and torque loading in the X direction

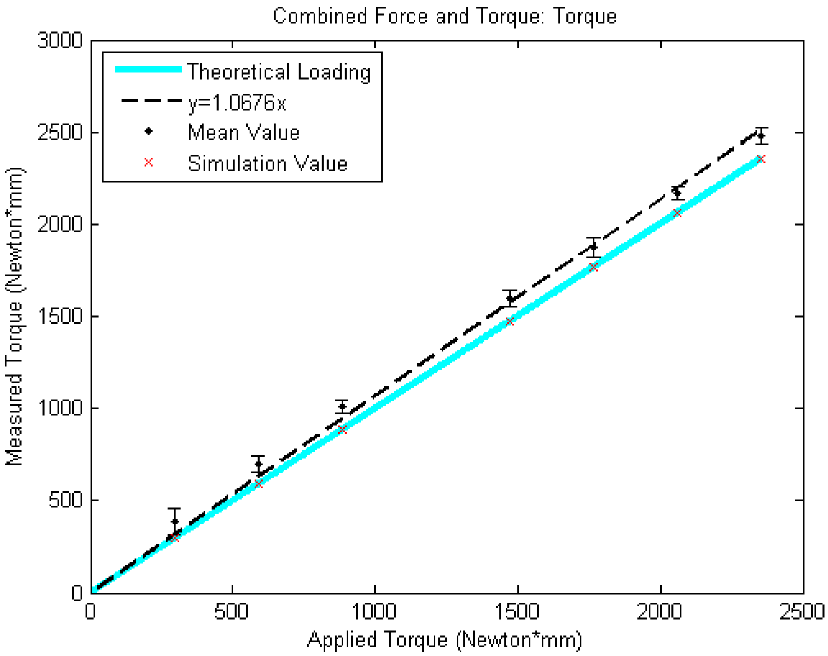


Figure 48: Measured and simulated torque for combined force and torque loading in the X direction

Table 8: Mean and Standard Deviation for Combined Force and Torque

Positive Fx and Positive Tz						Negative Fy and Negative Tz					
Fx (N)			Tz (Nmm)			Fy (N)			Tz (Nmm)		
Applied	Mean	SD	Applied	Mean	SD	Applied	Mean	SD	Applied	Mean	SD
9.81	14.23	0.56	294	388	73	-9.81	-11.13	1.13	294	-436	44
19.62	23.59	1.39	589	699	42	-19.62	-22.02	2.02	589	-711	50
29.43	33.14	2.00	883	1011	39	-29.43	-32.38	1.68	883	-981	64
49.05	52.05	0.90	1472	1600	45	-49.05	-51.30	1.65	1472	1571	27
58.86	61.92	1.29	1766	1876	51	-58.86	-62.26	1.89	1766	-1929	23
68.67	72.99	1.45	2060	2169	36	-68.67	-71.32	1.44	2060	-2157	76

4.5 Force at an angle

The last static test performed was the force at an angle test. This test only tested forces and not torques and set out to prove the ability of the device to effectively measure forces at an angle. The angles are all measured with theta equal to zero at the x axis for convention. This test was carried out with continuous loading, as carried out in the previous test. The results are comparable in linearity to testing the device without an angle offset. The results of this test are shown in Figure 49 and Figure 50. The mean and standard deviation are displayed in

Table 9: Mean and Standard Deviation for Force at an angle.

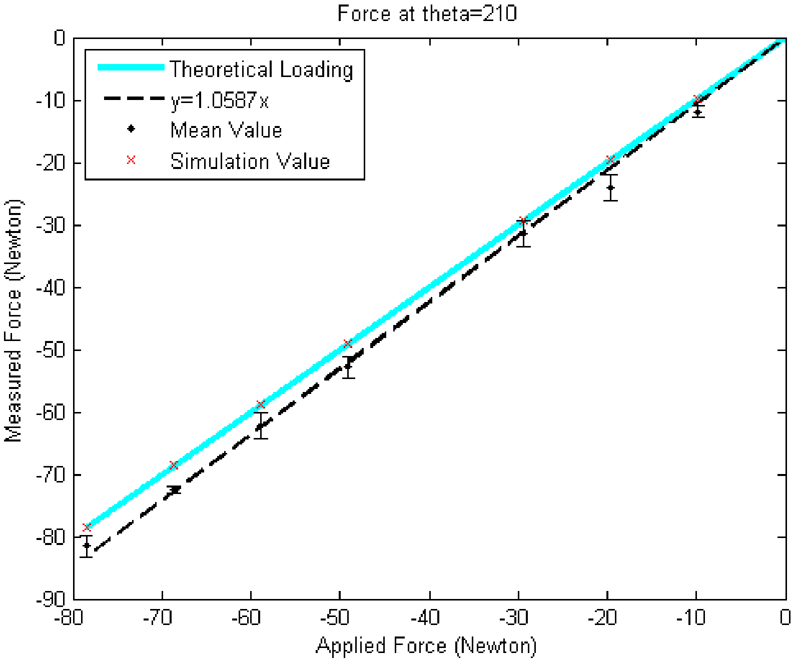


Figure 49: Measured and simulated loading in the Y direction at 210 degree

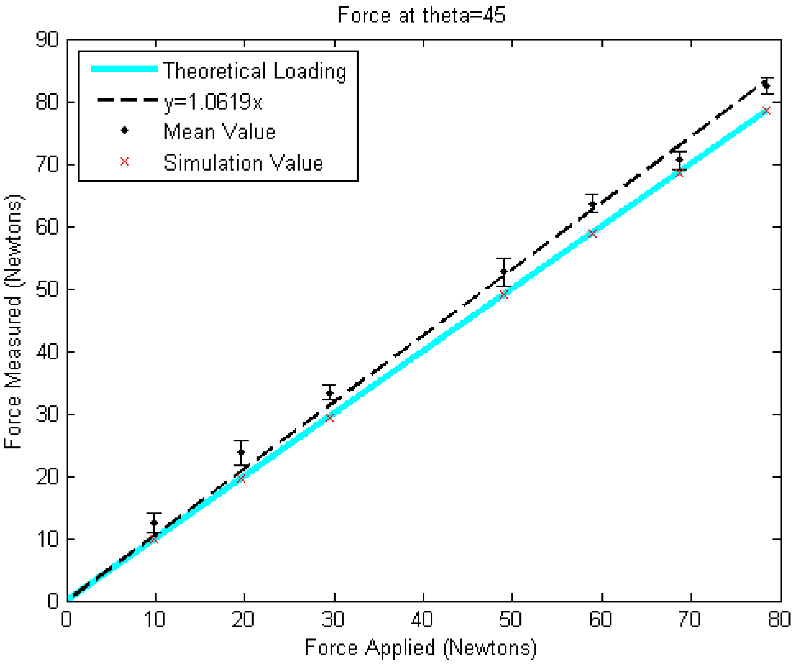


Figure 50: Measured and simulated loading in the Y direction at 45 degree

Table 9: Mean and Standard Deviation for Force at an angle

F at theta=210 (N)			F at theta=45 (N)		
Applied	Mean	SD	Applied	Mean	SD
-9.81	-11.93	0.88	9.81	12.51	1.66
-19.62	-24.04	2.07	19.62	23.75	1.94
-29.43	-31.55	2.10	29.43	33.32	1.19
-49.05	-52.88	1.63	49.05	52.09	2.26
-58.86	-62.32	2.19	58.86	63.64	1.50
-68.67	-72.48	0.51	68.67	70.56	1.43
-78.48	-81.57	1.75	78.48	83.34	1.34

CHAPTER 5

CONCLUSION

The study presented has successfully developed a planar, three degree of freedom force and torque sensor suitable for robotic applications. The device has the ability to accurately measure forces and torques in the X, Y, and rotation about the Z axis. The sensor described is accurate and repeatable, as displayed in the results. The low cost of the transducer designed produced a sensor which required a minimal amount of machining to produce.

Testing of the transducer revealed the capabilities of the device. Testing was carried out in logical manner, beginning with the most basic test which included symmetric loading to test how well the device output related to the theoretical output. From these tests, conclusions were drawn that the device was capable of accurately measuring static loading. The testing on the face of the device concluded that the device accurately and repeatably measures forces at the face of the device where a tool would be located. For each of the symmetric loading tests, the device was measured to be accurate and repeatable. The curve fits for the tests were within 1% of the theoretical value. For each of the tests where the face of the transducer was loaded, the deviation was within a 10% of the theoretical value.

Further research on this device can be broken up into two main sections, including further real world testing and developing a device capable of measuring 3 dimensional forces and torques, not just planar forces and torque. Real world testing still needs to be carried out in the actual use of this device on a robotic arm. Testing should be carried out in the future in a real world environment in order to verify this devices is usable in a manufacturing setting over an extended period of time. Because the experiments were undertaken in a static, controlled setting, the devices full performance has not been fully understood. Further testing will be the final verdict on whether this device has a future in the industry.

The study presented is the first step in successfully proving the ability of a force sensing resistor to measure planar forces and torques. While this study has only displayed a planar method, further research on this topic can be conducted to further this work to three dimensions. Using the design model provided in this paper, the equations can be extended to three dimensions. Thus, the groundwork for a three dimensional force-torque sensor using FSRs.

APPENDICES

APPENDIX A: Code for GUI

```
function varargout = Thesis_TrialGUI(varargin)
% THESIS_TRIALGUI MATLAB code for Thesis_TrialGUI.fig
%      THESIS_TRIALGUI, by itself, creates a new THESIS_TRIALGUI or
raises the existing
%      singleton*.
%
%      H = THESIS_TRIALGUI returns the handle to a new
THESIS_TRIALGUI or the handle to
%      the existing singleton*.
%
%      THESIS_TRIALGUI('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in THESIS_TRIALGUI.M with the given
input arguments.
%
%      THESIS_TRIALGUI('Property','Value',...) creates a new
THESIS_TRIALGUI or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before Thesis_TrialGUI_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Thesis_TrialGUI_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Thesis_TrialGUI

% Last Modified by GUIDE v2.5 20-May-2015 08:46:50

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @Thesis_TrialGUI_OpeningFcn, ...
    'gui_OutputFcn',  @Thesis_TrialGUI_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
```

```
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Thesis_TrialGUI is made visible.
function Thesis_TrialGUI_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Thesis_TrialGUI (see VARARGIN)

% Choose default command line output for Thesis_TrialGUI
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Thesis_TrialGUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);

global stopf
global s
global zsimflag
global func
global test
global f01
f01=zeros(4,1);
test=0;
s = serial('COM3');
stopf=1;
set(handles.btn_start,'visible','on')
set(handles.btn_exit,'visible','on')
set(handles.btn_stop,'visible','off')
set(handles.btn_save,'visible','off')
set(handles.btn_openfig,'visible','off')
zsimflag=0;
func = [-.707,.707,.707,-.707;-.707,.707,-.707,.707;12.065,12.065,-
12.065,-12.065];
func = [-.707,.707,.707,-.707;-.707,.707,-.707,.707;.012065,.012065,-
.012065,-.012065];
func = [-.707,.707,.707,-.707;-.707,.707,-
.707,.707;01.2065/2,01.2065/2,-01.2065/2,-01.2065/2];

% --- Outputs from this function are returned to the command line.
function varargout = Thesis_TrialGUI_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in btn_start.
function btn_start_Callback(hObject, eventdata, handles)
% hObject    handle to btn_start (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
clear allglobal
clear figures
global stopf
global test
stopf=0;
clear global f
clear global forc
clear global a
global s
global a
global t
global f
global forc
global c
global tf
global n
global f01
global tst
global f0
forc=zeros(3,1);
length(forc);
f=zeros(4,1);
global func
tst=get(handles.txt_savebx,'String');

pause(.1);
ser_start
set(handles.btn_start,'visible','off')
set(handles.btn_save,'visible','off')
set(handles.btn_openfig,'visible','off')
%% Set number of times to recieve data from Arduino
%%
n=1;
tfs=get(handles.txt_smp,'String');
tf=str2num(tfs);
while length(tfs)<6
    tfs=strcat('0',tfs);
end
t=(1:tf);
tp=t/200;
%% "Real Time" Plot data
%% Document the below lines better.... they are a bit confusing
b=500;                       %% b = number of samples to display
bd=15;                       %% bd = number of samples to
destroy/update
bi=b-1;                      %% bi = sample to start displaying from
```

```
uptxt=15;
%% Filter Variables
%% Current Filter is Moving Average Filter.... Try Others??
%% Filter Currently Does not do its job... Need to figure out Why!
Bl=25;
B = 1/Bl*ones(Bl,1);
pause(5); %% Pause to allow Arduino to "catch up"...... Note TIC
interfers
set(handles.btn_stop,'visible','on')
c=datestr(now,30);
tfo=t/100;
fwrite(s,tfs);
% tic;
k=10;
kl=6750
while stopf==0;
    stopf=1;
    for i=1:tf
        d(1,i) = str2double(fscanf(s));
        a(1,i) = str2double(fscanf(s));
        a(2,i) = str2double(fscanf(s));
        a(3,i) = str2double(fscanf(s));
        a(4,i) = str2double(fscanf(s));
        f(:,i)=  kl*(a(:,i)./(10*(1023-(a(:,i)))));
        f0(:,i)=f(:,i)-f01;
        forc(1:3,i)=func*f0(:,i);
        forc(4,i)=norm([forc(1,i),forc(2,i)]);
        if uptxt/i == 1
            out = filter(B,1,forc,[],2);

setforc=[str2num(get(handles.txt_fxa,'String'));str2num(get(handles.t
xt_fya,'String'));str2num(get(handles.txt_tza,'String'))];
            set(handles.tbl_sensor,'data', [a(:,i)';f0(:,i)']);
            set(handles.tbl_force,'data',
[[out(1:3,i)'];setforc';[out(1:3,i)-setforc]';100*[[out(1:3,i)-
setforc]'./setforc']]');%note '
            drawnow;
            uptxt=uptxt+50;
        end
       if test==0
        if i==130
            f01=mean(f(:,30:130),2);
            xlim([(b-bi) b]/200) ;
              f01=zeros(4,1);
        end
       end



        if  rem(i,b)== 0
            if (get(handles.rd_forcraw,'Value')) == 1
                guiplot=f0(:,b-bi:b);
            end
            if (get(handles.rd_transfilt,'Value')) == 1
                out = [filter(B,1,forc(1:3,:),[],2)];
                guiplot=out(:,b-bi:b);
            end
            drawnow;
            plot(handles.axes1,(t(1,b-bi):t(1,b))/200,guiplot);

            xlim([(b-bi) b]/200) ;
            b = b+bd;
        end

    end
end
% toc;
set(handles.btn_save,'visible','on')
set(handles.btn_start,'visible','on')
set(handles.btn_stop,'visible','off')
ser_stop
test=1;

% --- Executes when figure1 is resized.
function figure1_ResizeFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- Executes on button press in rd_transfilt.
function rd_transfilt_Callback(hObject, eventdata, handles)
% hObject    handle to rd_transfilt (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of rd_transfilt



% --- Executes on button press in rd_forcraw.
function rd_forcraw_Callback(hObject, eventdata, handles)
% hObject    handle to rd_forcraw (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of rd_forcraw



% --- Executes on button press in btn_save.
function btn_save_Callback(hObject, eventdata, handles)
% hObject    handle to btn_save (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;
global f0;
global forc;
global f
global fileName_jpg;
global fileName_fig;
global fl
global t;
global c
```

```
forc;
mx=length(forc);
ti=(0:(mx-400))/200;
ai=a(:,400:mx);
fi=f0(:,400:mx);
forci=forc(:,400:mx);
save_title;
save('Forcefile.mat','a','f','forc','t');

figure('Visible','off');
plot(ti,ai(1,:),ti,ai(2,:),ti,ai(3,:),ti,ai(4,:));
title('Sensor Digital Voltage Values');
xlabel('Time (s)');
ylabel('ADC (count)');
legend('FSR1','FSR2','FSR3','FSR4');
saveas(gcf,fileName_fig(1,:),'fig');

figure('Visible','off');
plot(ti,fi(:,:));
title('Sensor Force Values');
xlabel('Time (s)');
ylabel('Force (Newtons) Torque (N mm)');
legend('FSR1','FSR2','FSR3','FSR4');
saveas(gcf,fileName_fig(2,:),'fig');
% saveas(gcf,fileName_jpg(1,2),'-djpeg')

B = 1/40*ones(40,1);
for i=1:4;
    out(i,:) = filter(B,1,forc(i,:));
end;
figure('visible','off');
plot(ti,forci,ti,out(:,400:length(out)));
title('Sensor Digital Voltage Values');
xlabel('Time (s)');
ylabel('ADC (count)');
legend('Fx','Fy','Tz','Fmag','Fx','Fy','Tz','Fmag');
saveas(gcf,fileName_fig(3,:),'fig');

figure('Visible', 'off');
%
plot(t(Bl:tf),o(1,(Bl:tf)),t(Bl:tf),o(2,(Bl:tf)),t(Bl:tf),o(3,(Bl:tf)
),t(Bl:tf),o(4,(Bl:tf)));%,t,o(5,:))
plot(ti,out(:,400:length(out)))%,['c','b','r']);
title('Transducer Force Values');
xlabel('Time (s)');
ylabel(('Force (Newtons) Torque (N mm)'));
legend('Fx','Fy','Tz','Fmag');
saveas(gcf,fileName_fig(4,:),'fig');

figure('visible','off');
plot(ti,forci);    %%%%THIS SHOULD BE FORCF
title('Transducer Force Values');
xlabel('Time (s)');
ylabel('Force (Newtons) Torque (N mm)');
legend('Fx','Fy','Tz','Fmag');
```

```
saveas(gcf,fileName_fig(5,:),'fig'); %%%% CHECK THIS INDEX NUMBER>>>>
IT SHOULD BE 5

figure('visible','off');
subplot(211);
plot(ti,forci);    %%%%THIS SHOULD BE FORCF
title('Transducer Force Values');
xlabel('Time (s)');
ylabel('Force (Newtons) Torque (N mm)');
legend('Fx','Fy','Tz','Fmag');
subplot(212);
%
plot(t(Bl:tf),o(1,(Bl:tf)),t(Bl:tf),o(2,(Bl:tf)),t(Bl:tf),o(3,(Bl:tf)
),t(Bl:tf),o(4,(Bl:tf)));%,t,o(5,:))
plot(ti,fi(:,:));
title('Sensor Force Values');
xlabel('Time (s)');
ylabel('Force (Newtons) Torque (N mm)');
legend('FSR1','FSR2','FSR3','FSR4');
saveas(gcf,fileName_fig(6,:),'fig');

set(handles.btn_openfig,'visible','on')




% --- Executes on button press in btn_exit.
function btn_exit_Callback(hObject, eventdata, handles)
% hObject    handle to btn_exit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global s
if s~=0
    ser_stop
end
delete(handles.figure1)

% --- Executes on button press in btn_stop.
function btn_stop_Callback(hObject, eventdata, handles)
% hObject    handle to btn_stop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global stopf
global s
pause(1);
stopf=1;
ser_stop
set(handles.btn_start,'visible','on')
set(handles.btn_stop,'visible','off')
set(handles.btn_save,'visible','on')

function ser_stop
global s
global a
fclose(s)
delete(s)
```

```
s=0;

function ser_start
global s
s = serial('COM3');
%%Baud Rate must match Arduino Baud Rate
%% Formula for Baudrate to Sample = (Insert formula in the future)
global bri
br=9600*bri;
br=9600*6;
set(s,'BaudRate',br);
fopen(s);

function rb_sr_1_Callback(hObject, eventdata, handles)
% hObject    handle to rb_sr_1_Callback (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of rb_sr_1_Callback
global bri
bri=1

function rb_sr_2_Callback(hObject, eventdata, handles)
% hObject    handle to rb_sr_1_Callback (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of rb_sr_1_Callback
global bri
bri=2

function rb_sr_3_Callback(hObject, eventdata, handles)
% hObject    handle to rb_sr_3_Callback (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of rb_sr_3_Callback
global bri
bri=3

function rb_sr_4_Callback(hObject, eventdata, handles)
% hObject    handle to rb_sr_4_Callback (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of rb_sr_4_Callback
global bri
bri=4

function rb_sr_5_Callback(hObject, eventdata, handles)
% hObject    handle to rb_sr_5_Callback (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of rb_sr_5_Callback
```

```
global bri
bri=5

function save_title
global c
global fileName_fig
global y
global l
global tst
for k=1:9
    l=num2str(k);
%     xfig(k,:)=[strcat('C:\Users\pat\Desktop\Thesis\Testing
Results\','m',c,l,'_',tst,'.fig')];
    xfig(k,:)=[strcat('C:\Users\pat\Desktop\Thesis\Testing
Results\',tst,'_',l,'.fig')];
end
% fileName_jpg=xjpg;
fileName_fig=xfig;



% --- Executes on button press in btn_openfig.
function btn_openfig_Callback(hObject, eventdata, handles)
% hObject    handle to btn_openfig (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global t
global a
global forc
global f
global fileName_fig
for i=1:6
    openfig(fileName_fig(i,:),'visible');
end




function txt_smp_Callback(hObject, eventdata, handles)
% hObject    handle to txt_smp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of txt_smp as text
%        str2double(get(hObject,'String')) returns contents of
txt_smp as a double



% --- Executes during object creation, after setting all properties.
function txt_smp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to txt_smp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function txt_fxa_Callback(hObject, eventdata, handles)
% hObject    handle to txt_fxa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of txt_fxa as text
%        str2double(get(hObject,'String')) returns contents of
txt_fxa as a double


% --- Executes during object creation, after setting all properties.
function txt_fxa_CreateFcn(hObject, eventdata, handles)
% hObject    handle to txt_fxa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function txt_fya_Callback(hObject, eventdata, handles)
% hObject    handle to txt_fya (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of txt_fya as text
%        str2double(get(hObject,'String')) returns contents of
txt_fya as a double


% --- Executes during object creation, after setting all properties.
function txt_fya_CreateFcn(hObject, eventdata, handles)
% hObject    handle to txt_fya (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end



function txt_tza_Callback(hObject, eventdata, handles)
% hObject    handle to txt_tza (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of txt_tza as text
%        str2double(get(hObject,'String')) returns contents of
txt_tza as a double


% --- Executes during object creation, after setting all properties.
function txt_tza_CreateFcn(hObject, eventdata, handles)
% hObject    handle to txt_tza (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in btn_simulation.
function btn_simulation_Callback(hObject, eventdata, handles)
% hObject    handle to btn_simulation (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)




% --- Executes on button press in btn_zero.
function btn_zero_Callback(hObject, eventdata, handles)
% hObject    handle to btn_zero (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global forc
global f01
global f
m=length(f);
f01=mean(f(:,m-100:m),2);
% f01=zeros(4,1)

% --- Executes on button press in btn_simulate.
function btn_simulate_Callback(hObject, eventdata, handles)
% hObject    handle to btn_simulate (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global zsimflag
global forczerosim
global func
if zsimflag==0
    assignin('base','Fx',0)
    assignin('base','Fy',0)
    assignin('base','Tz',0)
    cd 'c:\Users\pat'
    evalin('base','Controls_Plant_3')
    % addpath('c:\Users\paSimulat\Desktop\Thesis\Simulation')
    sim('Planar_Simulation')
    digitalread=round(1023*47*S./(3188+47*S));
    S=3188*(digitalread./(47*(1023-digitalread)))
    forczerosim=func*S';
    zsimflag=1
end

assignin('base','Fx',str2num(get(handles.txt_fxa,'String')))
assignin('base','Fy',str2num(get(handles.txt_fya,'String')))
assignin('base','Tz',str2num(get(handles.txt_tza,'String')))
sim('Planar_Simulation')
digitalread=round(1023*47*S./(3188+47*S));
S=3188*(digitalread./(47*(1023-digitalread)))
forc=func*S'-forczerosim;
setforc=[str2num(get(handles.txt_fxa,'String'));str2num(get(handles.t
xt_fya,'String'));str2num(get(handles.txt_tza,'String'))];
set(handles.tbl_sensor,'data', [digitalread;S]);
set(handles.tbl_force,'data', [[forc(1:3)'];setforc';[forc(1:3)-
setforc]';100*[[forc(1:3)-setforc]'./setforc']]');%note '



function txt_savebx_Callback(hObject, eventdata, handles)
% hObject    handle to txt_savebx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of txt_savebx as text
%        str2double(get(hObject,'String')) returns contents of
txt_savebx as a double



% --- Executes during object creation, after setting all properties.
function txt_savebx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to txt_savebx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end

 '
```

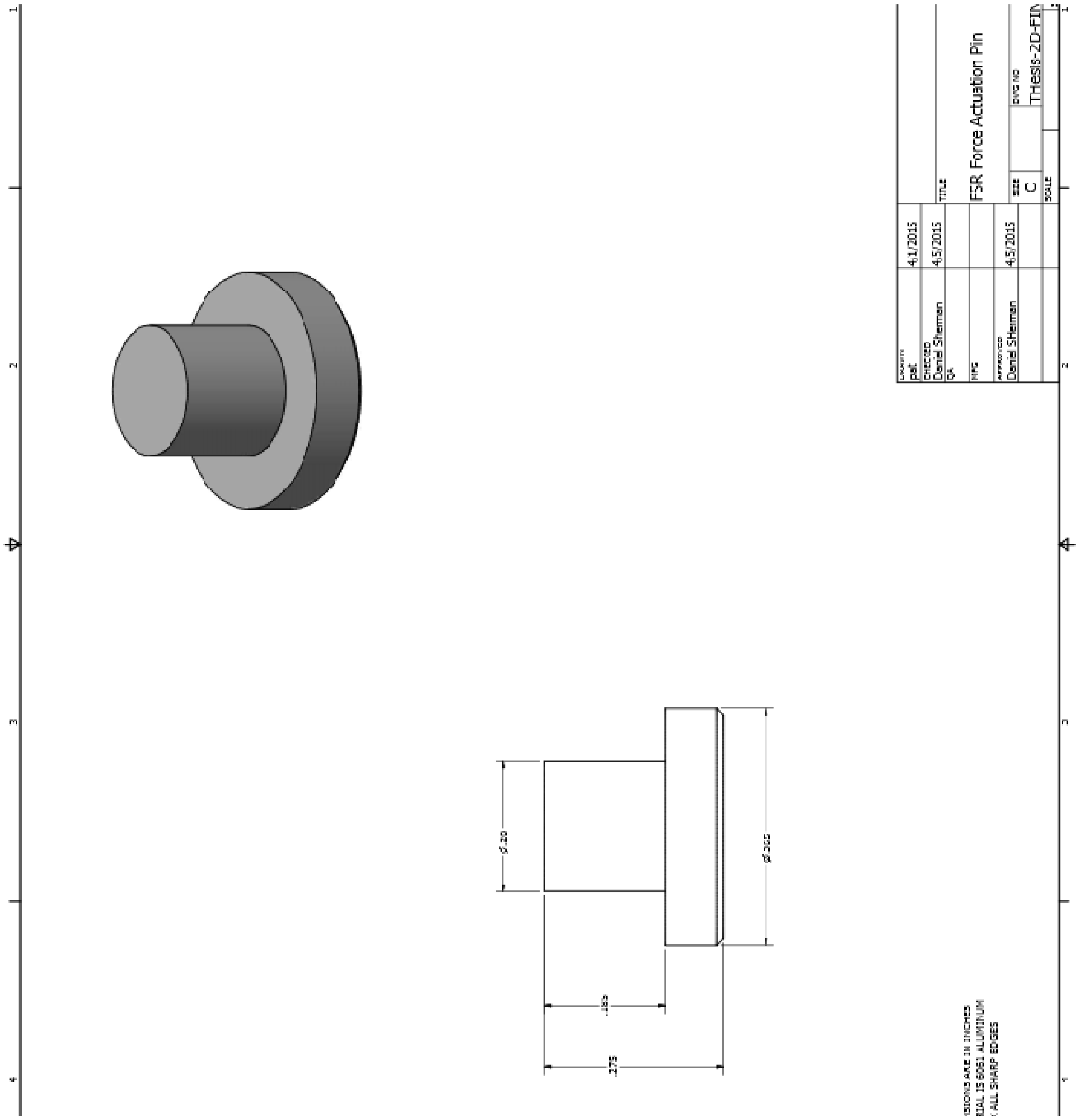
APPENDIX B: Arduino Code

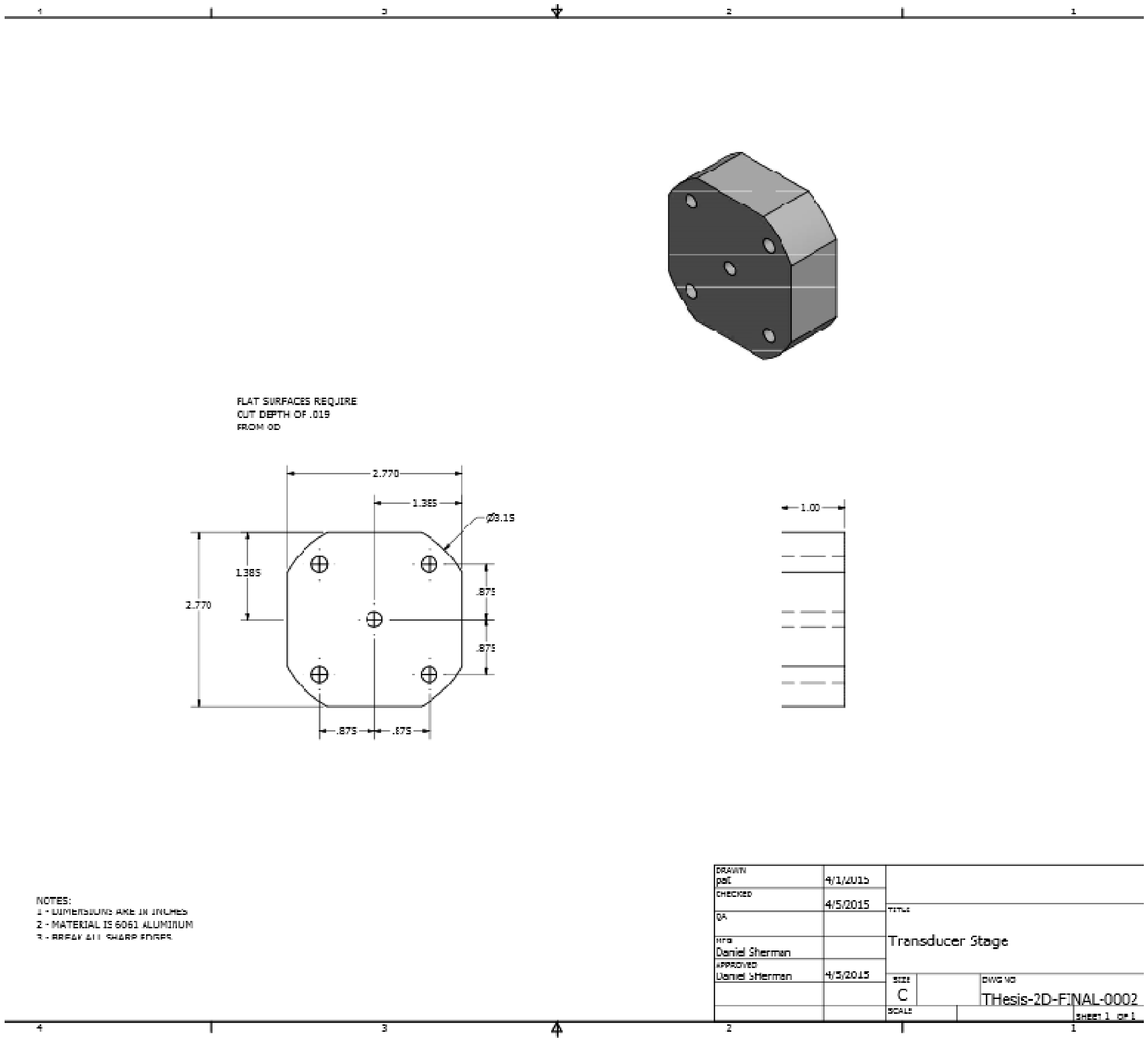
```
char serial[6];
char t[6];
int i=0;
unsigned int ti;
unsigned int tf;
int f;
int g;
void setup() {
  // put your setup code here, to run once:
//pinMode(13, OUTPUT);
Serial.begin(57600);
//pinMode(12, OUTPUT);

}

void loop() {
  // put your main code here, to run repeatedly
  while (i<6){
    while (Serial.available() > 0){
//      digitalWrite(12, HIGH);
      t[i] = Serial.read();
      i++;
    }
  }
  i=0;
  tf=atoi(t);
  ti=0;
//  Serial.println(tf);
    g=millis();
   while (ti<tf){
        int f=millis();
    if (f-g>5){
    Serial.println(f);
    Serial.println(analogRead(1));
    Serial.println(analogRead(2));
    Serial.println(analogRead(3));
   Serial.println(analogRead(4));
    ti++;
    g=g+5;
  }
 }
  if (strcmp(t, "1234")==0){
    Serial.println(2);
  }
//  digitalWrite(12, LOW);}
```

APPENDIX C: Technical Drawings



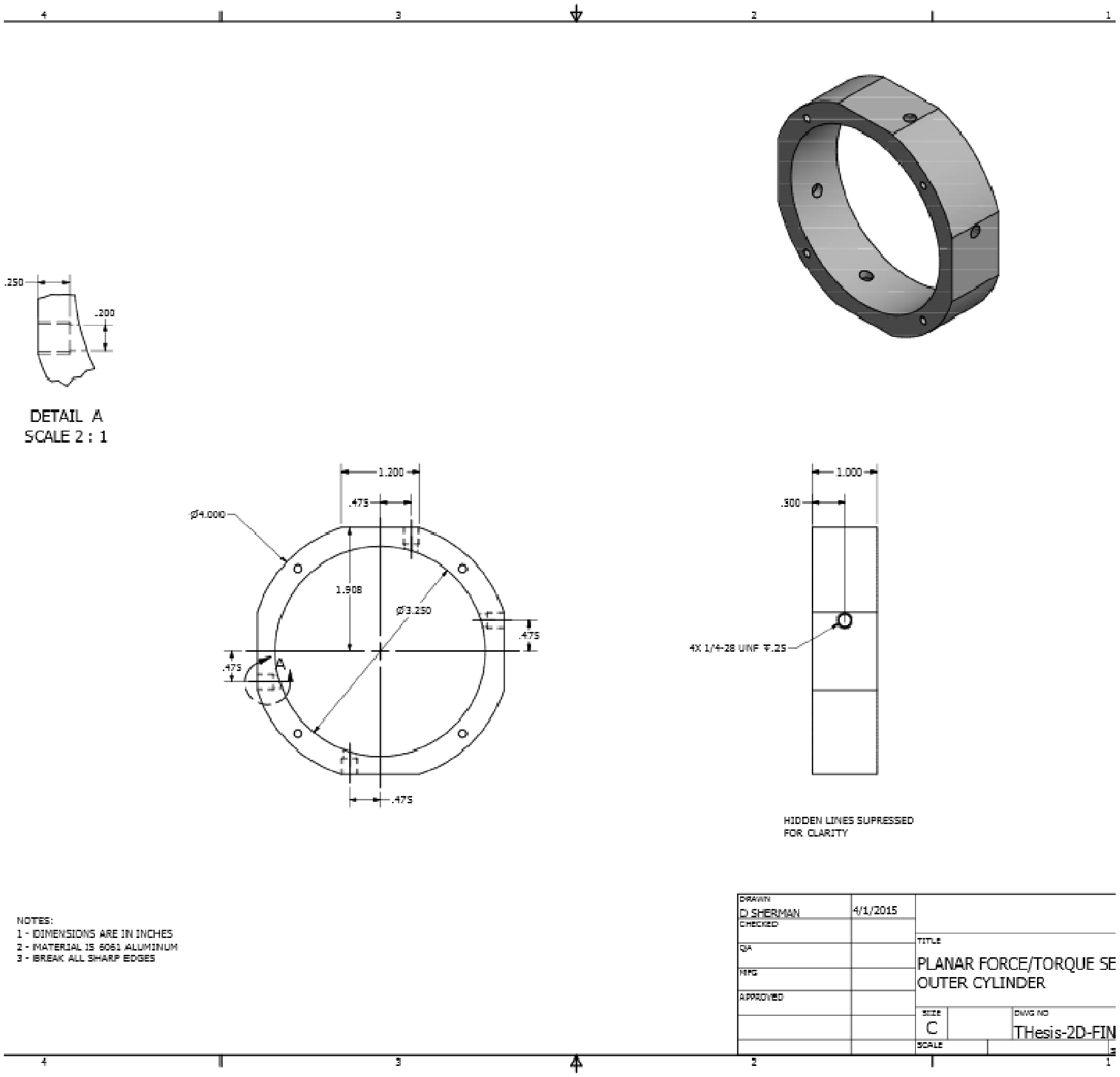


NOTES:
1 - DIMENSIONS ARE IN INCHES
2 - MATERIAL IS 6061 ALUMINUM
3 - BREAK ALL SHARP EDGES

DRAWN
pat 4/1/2015
CHECKED
4/5/2015
QA
MFG
Daniel Sherman
APPROVED
Daniel Sherman 4/5/2015

TITLE
Transducer Stage

SIZE
C
DWG NO
THesis-2D-FINAL-0002
SCALE
SHEET 1 OF 1



NOTES:
1 - IDIMENSIONS ARE IN INCHES
2 - IMATERIAL IS 6061 ALUMINUM
3 - IBREAK ALL SHARP EDGES

DRAWN D SHERMAN	4/1/2015	
CHECKED		TITLE PLANAR FORCE/TORQUE SE OUTER CYLINDER
QA		
MFG		
APPROVED		SIZE C · DWG NO THesis-2D-FIN
		SCALE

BBILIOGRAPHY

[1] Schumacher, Paul, and Jouaneh, Musa. "A Force Sensing Tool for Disassembly Operations." *Robotics and Computer-Integrated Manufacturing* 30.2 (2014): 206-17. Web.

[2] Park J, Kim G. Development of the 6-axis force/moment sensor for an intelligent robot's gripper. Sensors and Actuators A: Physical 2005; 118 (1):127–34.

[3] J. Fraden. Handbook of Modern Sensors. Physics, Design and Applications. Springer Verlag, 3rd edition, 2004.

[4] Tekscan Inc. FlexiForce A201 Standard Force & Load Sensors. Revision D, August 02, 2005.

[5] Eventoff, Franklin N. Electronic Pressure Sensitive Transducer Apparatus. Eventoff Franklin Neal, assignee. Patent 4314227. 2 Feb. 1982. Print.

[6] Jouaneh, Fundamentals of Mechatronics. USA: Cengage Learning, 2013

[7] Nikonovas A, Harrison AJL, Hoult S, Sammut D. The application of force-sensing resistor sensors for measuring forces developed by the human hand. Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine 2004; 218(2):121–6.

[8] Smith BT, Coiro DJ, Finson R, Betz RR, McCarthy J. Evaluation of force-sensing resistors for gait event detection to trigger electrical stimulation to improve walking in the child with cerebral palsy,. IEEE Transactions on Neural Systems and Rehabilitation Engineering 2002;10(1):22–9.M.

[9] Roland, Jeffrey Wyatt, Dylan John Seguin, and Richar Benjamin Knapp. Systems and Methods of Power Output Measurement. Roland Jeffrey Wyatt, Dylan John Seguin, Richard Benjamin Knapp, assignee. Patent US 8122773 B2. 28 Feb. 2012. Print.

[10] Baker, Jeffrey R. Pre-loaded Force Sensing Resistor and Method. Interlink Electronics, Inc., assignee. Patent 7772960. 10 Aug. 2010. Print.

[11] Nagasaka, Kenichiro and Yasunori Kawanami. Torque Sensor and Robot Apparatus. Sony Corporation, assignee. Patent US 8291775 B2. 23 Oct. 2012. Print.

[12] Su, Hai-Jun, and J. Michael Mccarthy. "A Polynomial Homotopy Formulation of the Inverse Static Analysis of Planar Compliant Mechanisms." Journal of Mechanical Design 128.4 (2006): 776-86.

[13] Roozbahani, Hamid, Alireza Fakhrizadeh, Heikki Haario, and Heikki Handroos. "Novel Online Re-Calibration Method for Multi-Axis Force/Torque Sensor of ITER Welding/Machining Robot." IEEE Sensors Journal 13.11 (2013): 4432-443. Web.

[14] Velásquez, Carlos Eduardo, and Rodríguez P. Yuli Andrea. 2010 IEEE ANDESCON Conference Proceedings 14-17 September 2010, Bogotá, Columbia. S.I.: IEEE, 2010. Print.